

# Advanced.h File Reference

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_depth_work_mode_list_count ob_depth_work_mode_list_get_count  
#define ob_device_preset_list_count ob_device_preset_list_get_count
```

## Functions

**OB\_EXPORT** **ob\_depth\_work\_mode** **ob\_device\_get\_current\_depth\_work\_mode** (const **ob\_device** \*device, **ob\_error** \*\*error)

Get the current depth work mode.

**OB\_EXPORT** const char \* **ob\_device\_get\_current\_depth\_work\_mode\_name** (const **ob\_device** \*device, **ob\_error** \*\*error)

Get current depth mode name.

**OB\_EXPORT** **ob\_status** **ob\_device\_switch\_depth\_work\_mode** (**ob\_device** \*device, const **ob\_depth\_work\_mode** \*work\_mode, **ob\_error** \*\*error)

Switch the depth work mode by **ob\_depth\_work\_mode**  
Prefer to use

**ob\_device\_switch\_depth\_work\_mode\_by\_name** to switch  
depth mode when the complete name of the depth work  
mode is known.

**OB\_EXPORT** **ob\_status** **ob\_device\_switch\_depth\_work\_mode\_by\_name**  
(**ob\_device** \*device, const char \*mode\_name, **ob\_error** \*\*error)

Switch the depth work mode by work mode name.

**OB\_EXPORT** **ob\_depth\_work\_mode\_list** \* **ob\_device\_get\_depth\_work\_mode\_list** (const **ob\_device** \*device, **ob\_error** \*\*error)

Request the list of supported depth work modes.

**OB\_EXPORT** uint32\_t **ob\_depth\_work\_mode\_list\_get\_count** (const **ob\_depth\_work\_mode\_list** \*work\_mode\_list, **ob\_error** \*\*error)

**OB\_EXPORT** **ob\_depth\_work\_mode** **ob\_depth\_work\_mode\_list\_get\_item** (const **ob\_depth\_work\_mode\_list** \*work\_mode\_list, uint32\_t index, **ob\_error** \*\*error)

Get the index target of **ob\_depth\_work\_mode** from work\_mode\_list.

**OB\_EXPORT** void **ob\_delete\_depth\_work\_mode\_list**  
(**ob\_depth\_work\_mode\_list** \*work\_mode\_list, **ob\_error** \*\*error)

Free the resources of **ob\_depth\_work\_mode\_list**.

**OB\_EXPORT** const char \* **ob\_device\_get\_current\_preset\_name** (const **ob\_device** \*device, **ob\_error** \*\*error)

The preset mean a set of parameters or configurations that can be applied to the device to achieve a specific effect or function.

**OB\_EXPORT** void **ob\_device\_load\_preset** (**ob\_device** \*device, const char \*preset\_name, **ob\_error** \*\*error)

Get the available preset list.

**OB\_EXPORT** void **ob\_device\_load\_preset\_from\_json\_file** (**ob\_device** \*device, const char \*json\_file\_path, **ob\_error** \*\*error)

Load preset from json string.

**OB\_EXPORT** void **ob\_device\_load\_preset\_from\_json\_data** (**ob\_device** \*device, const char \*presetName, const uint8\_t \*data, uint32\_t size, **ob\_error** \*\*error)

Load custom preset from data.

**OB\_EXPORT** void **ob\_device\_export\_current\_settings\_as\_preset\_json\_file** (**ob\_device** \*device, const char \*json\_file\_path, **ob\_error** \*\*error)

Export current settings as a preset json file.

**OB\_EXPORT** void **ob\_device\_export\_current\_settings\_as\_preset\_json\_data** (**ob\_device** \*device, const char \*presetName, const uint8\_t \*\*data, uint32\_t \*dataSize, **ob\_error** \*\*error)

Export current device settings as a preset json data.

**OB\_EXPORT** **ob\_device\_preset\_list** \* **ob\_device\_get\_available\_preset\_list** (const **ob\_device** \*device, **ob\_error** \*\*error)

Get the available preset list.

**OB\_EXPORT** void **ob\_delete\_preset\_list** (**ob\_device\_preset\_list** \*preset\_list, **ob\_error** \*\*error)

Delete the available preset list.

**OB\_EXPORT** uint32\_t **ob\_device\_preset\_list\_get\_count** (const **ob\_device\_preset\_list** \*preset\_list, **ob\_error** \*\*error)

Get the number of preset in the preset list.

**OB\_EXPORT** const char \* **ob\_device\_preset\_list\_get\_name** (const  
                 **ob\_device\_preset\_list** \*preset\_list, uint32\_t index,  
                 **ob\_error** \*\*error)

Get the name of the preset in the preset list.

**OB\_EXPORT** bool **ob\_device\_preset\_list\_has\_preset** (const  
                 **ob\_device\_preset\_list** \*preset\_list, const char  
                 \*preset\_name, **ob\_error** \*\*error)

Check if the preset list has the preset.

**OB\_EXPORT** bool **ob\_device\_is\_frame\_interleave\_supported** (const  
                 **ob\_device** \*device, **ob\_error** \*\*error)

Check if the device supports the frame interleave feature

**OB\_EXPORT** void **ob\_device\_load\_frame\_interleave** (**ob\_device** \*device,  
                 const char \*frame\_interleave\_name, **ob\_error** \*\*error)  
load the frame interleave mode according to frame  
interleavee name.

**OB\_EXPORT** **ob\_device\_frame\_interleave\_list** \* **ob\_device\_get\_available\_frame\_interleave\_list**  
(**ob\_device** \*device, **ob\_error** \*\*error)

Get the available frame interleave list.

**OB\_EXPORT** void **ob\_delete\_frame\_interleave\_list**  
(**ob\_device\_frame\_interleave\_list** \*frame\_interleave\_lis  
                 **ob\_error** \*\*error)

Delete the available frame interleave list.

**OB\_EXPORT** uint32\_t **ob\_device\_frame\_interleave\_list\_get\_count**  
(**ob\_device\_frame\_interleave\_list** \*frame\_interleave\_lis  
                 **ob\_error** \*\*error)

Get the number of frame interleave in the frame interleav  
list.

**OB\_EXPORT** const char \* **ob\_device\_frame\_interleave\_list\_get\_name**  
(**ob\_device\_frame\_interleave\_list** \*frame\_interleave\_lis  
                 uint32\_t index, **ob\_error** \*\*error)

Get the name of frame interleave in the frame interleave l

**OB\_EXPORT** bool **ob\_device\_frame\_interleave\_list\_has\_frame\_interlea**  
(**ob\_device\_frame\_interleave\_list** \*frame\_interleave\_lis  
                 const char \*frame\_interleave\_name, **ob\_error** \*\*error)

Check if the interleave ae list has the interleave ae.

**OB\_EXPORT** **ob\_preset\_resolution\_config\_list** \* **ob\_device\_get\_available\_preset\_resolution\_config\_list**  
(**ob\_device** \*device, **ob\_error** \*\*error)

**OB\_EXPORT** uint32\_t **ob\_device\_preset\_resolution\_config\_get\_count**  
(**ob\_preset\_resolution\_config\_list**  
                 \*bob\_preset\_resolution\_config\_list, **ob\_error** \*\*error)

```
OB_EXPORT OBPresetResolutionConfig ob_device_preset_resolution_config_list_get_item  
    (const ob_preset_resolution_config_list  
     *ob_preset_resolution_config_lis, uint32_t index, ob_error  
      **error)
```

Get the preset resolution in the preset resolution list.

```
OB_EXPORT void ob_delete_preset_resolution_config_list  
    (ob_preset_resolution_config_list  
     *ob_preset_resolution_config_list, ob_error **error)
```

Delete the available preset resolution list.

## Macro Definition Documentation

### ◆ ob\_depth\_work\_mode\_list\_count

```
#define ob_depth_work_mode_list_count ob_depth_work_mode_list_get_count
```

Definition at line **309** of file **Advanced.h**.

### ◆ ob\_device\_preset\_list\_count

```
#define ob_device_preset_list_count ob_device_preset_list_get_count
```

Definition at line **310** of file **Advanced.h**.

## Function Documentation

### ◆ ob\_device\_get\_current\_depth\_work\_mode()

```
OB_EXPORT ob_depth_work_mode ob_device_get_current_depth_work_mode ( const ob_device * device,
                                                               ob_error **      error )
```

Get the current depth work mode.

#### Parameters

- [in] **device** The device object.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_depth\_work\_mode** The current depth work mode.

Referenced by **ob::Device::getCurrentDepthWorkMode()**.

◆ **ob\_device\_get\_current\_depth\_work\_mode\_name()**

```
OB_EXPORT const char * ob_device_get_current_depth_work_mode_name ( const ob_device * device,
                                                               ob_error **      error )
```

Get current depth mode name.

According the current preset name to return current depth mode name

#### Returns

const char\* return the current depth mode name.

Referenced by **ob::Device::getCurrentDepthModeName()**.

◆ **ob\_device\_switch\_depth\_work\_mode()**

```
OB_EXPORT ob_status ob_device_switch_depth_work_mode ( ob_device * device,  
                                              const ob_depth_work_mode * work_mode,  
                                              ob_error ** error )
```

Switch the depth work mode by **ob\_depth\_work\_mode**. Prefer to use **ob\_device\_switch\_depth\_work\_mode\_by\_name** to switch depth mode when the complete name of the depth work mode is known.

### Parameters

- [in] **device** The device object.
- [in] **work\_mode** The depth work mode from **ob\_depth\_work\_mode\_list** which is returned by **ob\_device\_get\_depth\_work\_mode\_list**.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_status** The switch result. OB\_STATUS\_OK: success, other failed.

Referenced by **ob::Device::switchDepthWorkMode()**.

### ◆ **ob\_device\_switch\_depth\_work\_mode\_by\_name()**

```
OB_EXPORT ob_status ob_device_switch_depth_work_mode_by_name ( ob_device * device,  
                                              const char * mode_name,  
                                              ob_error ** error )
```

Switch the depth work mode by work mode name.

### Parameters

- [in] **device** The device object.
- [in] **mode\_name** The depth work mode name which is equal to **ob\_depth\_work\_mode.name**.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_status** The switch result. OB\_STATUS\_OK: success, other failed.

Referenced by **ob::Device::switchDepthWorkMode()**.

### ◆ **ob\_device\_get\_depth\_work\_mode\_list()**

Request the list of supported depth work modes.

## Parameters

[in] **device** The device object.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_depth\_work\_mode\_list** The list of **ob\_depth\_work\_mode**.

Referenced by [ob::Device::getDepthWorkModeList\(\)](#).

- #### ◆ ob\_depth\_work\_mode\_list\_get\_count()

## **OB\_EXPORT uint32\_t**

## ob\_depth\_work\_mode\_list\_get\_count

```
( const ob_depth_work_mode_list* work_mode_list,  
    ob_error** error )
```

Referenced by [ob::OBDepthWorkModeList::getCount\(\)](#).

- ◆ ob depth work mode list get item()

### **OB\_EXPORT ob\_depth\_work\_mode**

`ob_depth_work_mode_list_get_item`

```
( const ob_depth_work_mode_list* work_mode_list,  
    uint32_t                                index,  
    ob_error**                                error )
```

Get the index target of **ob depth work mode** from work mode list.

## Parameters

[in] **work\_mode\_list** Data structure containing a list of **ob depth work mode**

[in] **index** Index of the target **ob depth work mode**

**[out] error** Pointer to an error object that will be set if an error occurs.

## Returns

ob depth work mode

Referenced by [ob::OBDepthWorkModeList::getOBDepthWorkMode\(\)](#).

◆ [ob\\_delete\\_depth\\_work\\_mode\\_list\(\)](#)

```
OB_EXPORT void ob_delete_depth_work_mode_list ( ob_depth_work_mode_list * work_mode_list,  
                                              ob_error **          error )
```

Free the resources of [ob\\_depth\\_work\\_mode\\_list](#).

**Parameters**

[in] **work\_mode\_list** Data structure containing a list of [ob\\_depth\\_work\\_mode](#)  
[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::OBDepthWorkModeList::~OBDepthWorkModeList\(\)](#).

◆ [ob\\_device\\_get\\_current\\_preset\\_name\(\)](#)

```
OB_EXPORT const char * ob_device_get_current_preset_name ( const ob_device * device,  
                                                       ob_error **      error )
```

The preset mean a set of parameters or configurations that can be applied to the device to achieve a specific effect or function.

@breif Get the current preset name.

**Parameters**

**device** The device object.  
**error** Pointer to an error object that will be set if an error occurs.

**Returns**

The current preset name, it should be one of the preset names returned by [ob\\_device\\_get\\_available\\_preset\\_list](#).

Referenced by [ob::Device::getCurrentPresetName\(\)](#).

◆ [ob\\_device\\_load\\_preset\(\)](#)

```
OB_EXPORT void ob_device_load_preset ( ob_device * device,  
                                     const char * preset_name,  
                                     ob_error ** error )
```

Get the available preset list.

### Attention

After loading the preset, the settings in the preset will set to the device immediately. Therefore, it is recommended to re-read the device settings to update the user program temporarily.

### Parameters

**device** The device object.  
**preset\_name** Pointer to an error object that will be set if an error occurs. The name should be one of the preset names returned by [ob\\_device\\_get\\_available\\_preset\\_list](#).  
**error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::loadPreset\(\)](#).

### ◆ [ob\\_device\\_load\\_preset\\_from\\_json\\_file\(\)](#)

```
OB_EXPORT void ob_device_load_preset_from_json_file ( ob_device * device,  
                                         const char * json_file_path,  
                                         ob_error ** error )
```

Load preset from json string.

After loading the custom preset, the settings in the custom preset will set to the device immediately.

After loading the custom preset, the available preset list will be appended with the custom preset and named as the file name.

### Parameters

**device** The device object.  
**json\_file\_path** The json file path.  
**error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::loadPresetFromFile\(\)](#).

### ◆ [ob\\_device\\_load\\_preset\\_from\\_json\\_data\(\)](#)

```
OB_EXPORT void ob_device_load_preset_from_json_data ( ob_device * device,  
                                                 const char * presetName,  
                                                 const uint8_t * data,  
                                                 uint32_t size,  
                                                 ob_error ** error )
```

Load custom preset from data.

After loading the custom preset, the settings in the custom preset will set to the device immediately.

After loading the custom preset, the available preset list will be appended with the custom preset and named as the presetName.

### Attention

The user should ensure that the custom preset data is adapted to the device and the settings in the data are valid.

It is recommended to re-read the device settings to update the user program temporarily after successfully loading the custom preset.

### Parameters

**data** The custom preset data.

**size** The size of the custom preset data.

Referenced by **ob::Device::loadPresetFromJsonData()**.

- ◆ [ob\\_device\\_export\\_current\\_settings\\_as\\_preset\\_json\\_file\(\)](#)

Export current settings as a preset json file.

After exporting the custom preset, the available preset list will be appended with the custom preset and named as the file name.

## Parameters

**device** The device object.

**json\_file\_path** The json file path.

**error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::exportSettingsAsPresetJsonFile\(\)](#).

- ◆ ob\_device\_export\_current\_settings\_as\_preset\_json\_data()

Export current device settings as a preset json data.

After exporting the preset, a new preset named as the presetName will be added to the available preset list.

## Attention

The memory of the data is allocated by the SDK, and will automatically be released by the SDK.

The memory of the data will be reused by the SDK on the next call, so the user should copy the data to a new buffer if it needs to be preserved.

## Parameters

[out] **data** return the preset json data.

[out] **dataSize** return the size of the preset json data.

Referenced by [ob::Device::exportSettingsAsPresetJsonData\(\)](#).

- #### ◆ ob\_device\_get\_available\_preset\_list()

Get the available preset list.

## Parameters

**device** The device object.

**error** Pointer to an error object that will be set if an error occurs.

## Returns

The available preset list.

Referenced by [ob::Device::getAvailablePresetList\(\)](#).

- #### ◆ ob\_delete\_preset\_list()

Delete the available preset list.

## Parameters

**preset\_list** The available preset list.

**error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::DevicePresetList::~DevicePresetList\(\)](#).

- #### ◆ ob\_device\_preset\_list\_get\_count()

```
OB_EXPORT uint32_t ob_device_preset_list_get_count ( const ob_device_preset_list* preset_list,  
                                                 ob_error **  
                                                 error )
```

Get the number of preset in the preset list.

#### Parameters

**preset\_list** The available preset list.  
**error** Pointer to an error object that will be set if an error occurs.

#### Returns

The number of preset in the preset list.

Referenced by **ob::DevicePresetList::getCount()**.

### ◆ **ob\_device\_preset\_list\_get\_name()**

```
OB_EXPORT const char * ob_device_preset_list_get_name ( const ob_device_preset_list* preset_list,  
                                                       uint32_t  
                                                       index,  
                                                       ob_error **  
                                                       error )
```

Get the name of the preset in the preset list.

#### Parameters

**preset\_list** The available preset list.  
**index** The index of the preset in the preset list.  
**error** Pointer to an error object that will be set if an error occurs.

#### Returns

The name of the preset in the preset list.

Referenced by **ob::DevicePresetList::getName()**.

### ◆ **ob\_device\_preset\_list\_has\_preset()**

```
OB_EXPORT bool ob_device_preset_list_has_preset ( const ob_device_preset_list* preset_list,  
                                              const char* preset_name,  
                                              ob_error** error )
```

Check if the preset list has the preset.

#### Parameters

**preset\_list** The available preset list.

**preset\_name** The name of the preset.

**error** Pointer to an error object that will be set if an error occurs.

#### Returns

Whether the preset list has the preset. If true, the preset list has the preset. If false, the preset list does not have the preset.

Referenced by [ob::DevicePresetList::hasPreset\(\)](#).

### ◆ [ob\\_device\\_is\\_frame\\_interleave\\_supported\(\)](#)

```
OB_EXPORT bool ob_device_is_frame_interleave_supported ( const ob_device* device,  
                                                       ob_error** error )
```

Check if the device supports the frame interleave feature.

#### Parameters

**device** The device object.

**error** Pointer to an error object that will be set if an error occurs.

#### Returns

bool Returns true if the device supports the frame interleave feature.

Referenced by [ob::Device::isFrameInterleaveSupported\(\)](#).

### ◆ [ob\\_device\\_load\\_frame\\_interleave\(\)](#)

```
OB_EXPORT void ob_device_load_frame_interleave ( ob_device * device,  
                                              const char * frame_interleave_name,  
                                              ob_error ** error )
```

load the frame interleave mode according to frame interleavee name.

#### Parameters

**device** The device object.  
**frame\_interleave\_name** The name should be one of the frame interleave names returned by  
[\*\*ob\\_device\\_get\\_available\\_frame\\_interleave\\_list\*\*](#).  
**error** Log error messages.

Referenced by [\*\*ob::Device::loadFrameInterleave\(\)\*\*](#).

◆ [\*\*ob\\_device\\_get\\_available\\_frame\\_interleave\\_list\(\)\*\*](#)

```
OB_EXPORT ob_device_frame_interleave_list *  
ob_device_get_available_frame_interleave_list  
                                ( ob_device * device,  
                                ob_error ** error )
```

Get the available frame interleave list.

#### Parameters

**device** The device object.  
**error** Log error messages.

#### Returns

The available frame interleave list.

Referenced by [\*\*ob::Device::getAvailableFrameInterleaveList\(\)\*\*](#).

◆ [\*\*ob\\_delete\\_frame\\_interleave\\_list\(\)\*\*](#)

```
OB_EXPORT void ob_delete_frame_interleave_list ( ob_device_frame_interleave_list * frame_interleave_list,  
                                              ob_error **  
                                              error )
```

Delete the available frame interleave list.

#### Parameters

**frame\_interleave\_list** The available frame interleave list.  
**error** Log error messages.

Referenced by **ob::DeviceFrameInterleaveList::~DeviceFrameInterleaveList()**.

#### ◆ **ob\_device\_frame\_interleave\_list\_get\_count()**

```
OB_EXPORT uint32_t  
ob_device_frame_interleave_list_get_count ( ob_device_frame_interleave_list * frame_interleave_list,  
                                            ob_error **  
                                            error )
```

Get the number of frame interleave in the frame interleave list.

#### Parameters

**frame\_interleave\_list** The available frame interleave list.  
**error** Log error messages.

#### Returns

The number of frame interleave in the frame interleave list.

Referenced by **ob::DeviceFrameInterleaveList::getCount()**.

#### ◆ **ob\_device\_frame\_interleave\_list\_get\_name()**

```

OB_EXPORT const char *
ob_device_frame_interleave_list_get_name( ob_device_frame_interleave_list * frame_interleave_list,
                                         uint32_t index,
                                         ob_error ** error )

```

Get the name of frame interleave in the frame interleave list.

#### Parameters

<b>frame_interleave_list</b>	The available frame interleave list.
<b>index</b>	The index of frame interleave in the frame interleave list.
<b>error</b>	Log error messages.

#### Returns

The name of frame interleave in the frame interleave list..

Referenced by [ob::DeviceFrameInterleaveList::getName\(\)](#).

- ◆ [ob\\_device\\_frame\\_interleave\\_list\\_has\\_frame\\_interleave\(\)](#)

```

OB_EXPORT bool
ob_device_frame_interleave_list_has_frame_interleave( ob_device_frame_interleave_list * frame_interleave_list,
                                                       const char * frame_interleave_name,
                                                       ob_error ** error )

```

Check if the interleave ae list has the interleave ae.

#### Parameters

<b>frame_interleave_list</b>	The available interleave ae list.
<b>frame_interleave_name</b>	The name of the interleave ae.
<b>error</b>	Log error messages.

#### Returns

Whether the interleave ae list has the interleave ae. If true, the interleave ae list has the interleave ae.  
If false, the interleave ae list does not have the interleave ae.

Referenced by [ob::DeviceFrameInterleaveList::hasFrameInterleave\(\)](#).

- ◆ [ob\\_device\\_get\\_available\\_preset\\_resolution\\_config\\_list\(\)](#)

```
OB_EXPORT ob_preset_resolution_config_list *
ob_device_get_available_preset_resolution_config_list
( ob_device * device,
  ob_error ** error )
```

Referenced by **ob::Device::getAvailablePresetResolutionConfigList()**.

◆ **ob\_device\_preset\_resolution\_config\_get\_count()**

```
OB_EXPORT uint32_t
ob_device_preset_resolution_config_get_count ( ob_preset_resolution_config_list * ob_preset_resolution_config_
                                              ob_error ** error )
```

Referenced by **ob::PresetResolutionConfigList::getCount()**.

◆ **ob\_device\_preset\_resolution\_config\_list\_get\_item()**

```
OB_EXPORT OBPresetResolutionConfig
ob_device_preset_resolution_config_list_get_item ( const ob_preset_resolution_config_list * ob_preset_resolution
                                                 uint32_t
                                                 index,
                                                 ob_error ** error )
```

Get the preset resolution in the preset resolution list.

#### Parameters

<b>ob_preset_resolution_config_list</b>	The available preset resolution list.
<b>index</b>	The index of preset resolution in the preset resolution list.
<b>error</b>	Log error messages.

#### Returns

The preset resolution in the preset resolution list.

Referenced by **ob::PresetResolutionConfigList::getPresetResolutionRatioConfig()**.

◆ **ob\_delete\_preset\_resolution\_config\_list()**

```
OB_EXPORT void  
ob_delete_preset_resolution_config_list ( ob_preset_resolution_config_list* ob_preset_resolution_config_list,  
                                         ob_error ***  
                                         error )
```

Delete the available preset resolution list.

### Parameters

**frame\_interleave\_list** The available preset resolution list.  
**error** Log error messages.

Referenced by [ob::PresetResolutionConfigList::~PresetResolutionConfigList\(\)](#).

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# Advanced.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #ifndef __cplusplus
5 extern "C" {
6 #endif
7
8 #include "ObTypes.h"
9
18 OB_EXPORT ob_depth_work_mode ob_device_get_current_depth_work_mode(const ob_device *device
19 , ob_error **error);
25 OB_EXPORT const char *ob_device_get_current_depth_work_mode_name(const ob_device *device, ob
26 _error **error);
37 OB_EXPORT ob_status ob_device_switch_depth_work_mode(ob_device *device, const ob_depth_work
38 _mode *work_mode, ob_error **error);
48 OB_EXPORT ob_status ob_device_switch_depth_work_mode_by_name(ob_device *device, const char *
49 mode_name, ob_error **error);
58 OB_EXPORT ob_depth_work_mode_list *ob_device_get_depth_work_mode_list(const ob_device *devic
59 e, ob_error **error);
68 OB_EXPORT uint32_t ob_depth_work_mode_list_get_count(const ob_depth_work_mode_list *work_mo
69 de_list, ob_error **error);
79 OB_EXPORT ob_depth_work_mode ob_depth_work_mode_list_get_item(const ob_depth_work_mode_li
80 st *work_mode_list, uint32_t index, ob_error **error);
88 OB_EXPORT void ob_delete_depth_work_mode_list(ob_depth_work_mode_list *work_mode_list, ob_err
89 or **error);
98 OB_EXPORT const char *ob_device_get_current_preset_name(const ob_device *device, ob_error **erro
90 r);
110 OB_EXPORT void ob_device_load_preset(ob_device *device, const char *preset_name, ob_error **erro
111 r);
121 OB_EXPORT void ob_device_load_preset_from_json_file(ob_device *device, const char *json_file_path
122 , ob_error **error);
134 OB_EXPORT void ob_device_load_preset_from_json_data(ob_device *device, const char *presetName,
135 const uint8_t *data, uint32_t size, ob_error **error);
144 OB_EXPORT void ob_device_export_current_settings_as_preset_json_file(ob_device *device, const cha
145 r *json_file_path, ob_error **error);
157 OB_EXPORT void ob_device_export_current_settings_as_preset_json_data(ob_device *device, const c
158 har *presetName, const uint8_t **data, uint32_t *dataSize,
159 ob_error **error);
167 OB_EXPORT ob_device_preset_list *ob_device_get_available_preset_list(const ob_device *device, ob_
168 error **error);
175 OB_EXPORT void ob_delete_preset_list(ob_device_preset_list *preset_list, ob_error **error);
176
184 OB_EXPORT uint32_t ob_device_preset_list_get_count(const ob_device_preset_list *preset_list, ob_err
```

```

185     - - - - - or **error);
194 OB_EXPORT const char *ob_device_preset_list_get_name(const ob_device_preset_list *preset_list, uint
195         32_t index, ob_error **error);
196
204 OB_EXPORT bool ob_device_preset_list_has_preset(const ob_device_preset_list *preset_list, const cha
205         r *preset_name, ob_error **error);
206
213 OB_EXPORT bool ob_device_is_frame_interleave_supported(const ob_device *device, ob_error **error)
214         ;
222 OB_EXPORT void ob_device_load_frame_interleave(ob_device *device, const char *frame_interleave_na
223         me, ob_error **error);
224
231 OB_EXPORT ob_device_frame_interleave_list *ob_device_get_available_frame_interleave_list(ob_devic
232         e *device, ob_error **error);
233
239 OB_EXPORT void ob_delete_frame_interleave_list(ob_device_frame_interleave_list *frame_interleave_li
234         st, ob_error **error);
235
248 OB_EXPORT uint32_t ob_device_frame_interleave_list_get_count(ob_device_frame_interleave_list *fra
249         me_interleave_list, ob_error **error);
250
258 OB_EXPORT const char *ob_device_frame_interleave_list_get_name(ob_device_frame_interleave_list *f
259         rame_interleave_list, uint32_t index, ob_error **error);
260
269 OB_EXPORT bool ob_device_frame_interleave_list_has_frame_interleave(ob_device_frame_interleave_li
270         st *frame_interleave_list, const char *frame_interleave_name,
271             ob_error **error);
272
273 /* @brief Get the available preset resolution config list.
274 */
275 /* @param device The device object.
276 */
277 /* @param error Log error messages.
278 */
279 /* @return The available frame interleave list.
280 */
281 OB_EXPORT ob_preset_resolution_config_list *ob_device_get_available_preset_resolution_config_list(
282         ob_device *device, ob_error **error);
283
284 /* @brief Get the number of preset resolution in the preset resolution list.
285 */
286 /* @param ob_preset_resolution_config_list The available preset resolution list.
287 */
288 /* @param error Log error messages.
289 */
290 /* @return The number of preset resolution in the preset resolution list.
291 */
292 OB_EXPORT uint32_t ob_device_preset_resolution_config_get_count(ob_preset_resolution_config_list
293         *ob_preset_resolution_config_list, ob_error **error);
294
295 OB_EXPORT OBPresetResolutionConfig ob_device_preset_resolution_config_list_get_item(const ob_pr
296         eset_resolution_config_list *ob_preset_resolution_config_list,
297             uint32_t index, ob_error **error);
298
299 OB_EXPORT void ob_delete_preset_resolution_config_list(ob_preset_resolution_config_list *ob_preset
300         _resolution_config_list, ob_error **error);
301
302 // The following interfaces are deprecated and are retained here for compatibility purposes.
303 #define ob_depth_work_mode_list_count ob_depth_work_mode_list_get_count
304 #define ob_device_preset_list_count ob_device_preset_list_get_count
305
306 #ifndef __cplusplus
307 }
308 #endif

```



# Context.h File Reference

Context is a management class that describes the runtime of the SDK and is responsible for resource allocation and release of the SDK. Context has the ability to manage multiple devices. It is responsible for enumerating devices, monitoring device callbacks, and enabling multi-device synchronization. [More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_enable_multi_device_sync ob_enable_device_clock_sync  
#define ob_set_logger_callback ob_set_logger_to_callback
```

## Functions

```
OB_EXPORT ob_context * ob_create_context (ob_error **error)  
Create a context object with the default configuration file.  
  
OB_EXPORT ob_context * ob_create_context_with_config (const char *config_file_path,  
ob_error **error)  
Create a context object with a specified configuration file.  
  
OB_EXPORT void ob_delete_context (ob_context *context, ob_error **error)  
Delete a context object.  
  
OB_EXPORT ob_device_list * ob_query_device_list (ob_context *context, ob_error **error)  
Get a list of enumerated devices.  
  
OB_EXPORT void ob_enable_net_device_enumeration (ob_context *context, bool  
enable, ob_error **error)  
Enable or disable network device enumeration.  
  
OB_EXPORT ob_device * ob_create_net_device (ob_context *context, const char *address,  
uint16_t port, ob_error **error)  
Create a network device object.  
  
OB_EXPORT void ob_set_device_changed_callback (ob_context *context,  
ob_device_changed_callback callback, void *user_data, ob_error  
**error)  
Set a device plug-in callback function.  
  
OB_EXPORT void ob_enable_device_clock_sync (ob_context *context, uint64_t  
repeat_interval_msec, ob_error **error)  
Activates device clock synchronization to synchronize the clock of the  
host and all created devices (if supported).
```

**OB\_EXPORT** void **ob\_free\_idle\_memory** (**ob\_context** \*context, **ob\_error** \*\*error)

Free idle memory from the internal frame memory pool.

**OB\_EXPORT** void **ob\_set\_uvc\_backend\_type** (**ob\_context** \*context,

**ob\_uvc\_backend\_type** backend\_type, **ob\_error** \*\*error)

For linux, there are two ways to enable the UVC backend: libuvc and v4l2. This function is used to set the backend type.

**OB\_EXPORT** void **ob\_set\_logger\_severity** (**ob\_log\_severity** severity, **ob\_error** \*\*error)

Set the global log level.

**OB\_EXPORT** void **ob\_set\_logger\_to\_file** (**ob\_log\_severity** severity, const char \*directory,

**ob\_error** \*\*error)

Set the log output to a file.

**OB\_EXPORT** void **ob\_set\_logger\_to\_callback** (**ob\_log\_severity** severity, **ob\_log\_callback**

callback, void \*user\_data, **ob\_error** \*\*error)

Set the log callback function.

**OB\_EXPORT** void **ob\_set\_logger\_to\_console** (**ob\_log\_severity** severity, **ob\_error**

\*\*error)

Set the log output to the console.

**OB\_EXPORT** void **ob\_set\_extensions\_directory** (const char \*directory, **ob\_error** \*\*error)

Set the extensions directory.

## Detailed Description

Context is a management class that describes the runtime of the SDK and is responsible for resource allocation and release of the SDK. Context has the ability to manage multiple devices. It is responsible for enumerating devices, monitoring device callbacks, and enabling multi-device synchronization.

Definition in file [Context.h](#).

## Macro Definition Documentation

### ◆ [ob\\_enable\\_multi\\_device\\_sync](#)

```
#define ob_enable_multi_device_sync ob\_enable\_device\_clock\_sync
```

Definition at line [168](#) of file [Context.h](#).

### ◆ [ob\\_set\\_logger\\_callback](#)

```
#define ob_set_logger_callback ob_set_logger_to_callback
```

Definition at line **169** of file **Context.h**.

## Function Documentation

### ◆ **ob\_create\_context()**

```
OB_EXPORT ob_context * ob_create_context ( ob_error ** error )
```

Create a context object with the default configuration file.

#### Parameters

[out] **error** Pointer to an error object that will be populated if an error occurs during context creation

#### Returns

Pointer to the created context object

### ◆ **ob\_create\_context\_with\_config()**

```
OB_EXPORT ob_context * ob_create_context_with_config ( const char * config_file_path,  
                                                    ob_error ** error )
```

Create a context object with a specified configuration file.

#### Parameters

[in] **config\_file\_path** Path to the configuration file. If NULL, the default configuration file will be used.

[out] **error** Pointer to an error object that will be populated if an error occurs during context creation

#### Returns

Pointer to the created context object

Referenced by **ob::Context::Context()**.

### ◆ **ob\_delete\_context()**

```
OB_EXPORT void ob_delete_context ( ob_context* context,  
                                    ob_error ** error )
```

Delete a context object.

#### Parameters

- [in] **context** Pointer to the context object to be deleted
- [out] **error** Pointer to an error object that will be populated if an error occurs during context deletion

Referenced by [ob::Context::~Context\(\)](#).

#### ◆ [ob\\_query\\_device\\_list\(\)](#)

```
OB_EXPORT ob_device_list * ob_query_device_list ( ob_context* context,  
                                              ob_error ** error )
```

Get a list of enumerated devices.

#### Parameters

- [in] **context** Pointer to the context object
- [out] **error** Pointer to an error object that will be populated if an error occurs during device enumeration

#### Returns

Pointer to the device list object

Referenced by [ob::Context::queryDeviceList\(\)](#).

#### ◆ [ob\\_enable\\_net\\_device\\_enumeration\(\)](#)

```
OB_EXPORT void ob_enable_net_device_enumeration ( ob_context* context,  
                                              bool        enable,  
                                              ob_error ** error )
```

Enable or disable network device enumeration.

After enabling, the network device will be automatically discovered and can be retrieved through [ob\\_query\\_device\\_list](#). The default state can be set in the configuration file.

### Attention

Network device enumeration is performed through the GVCP protocol. If the device is not in the same subnet as the host, it will be discovered but cannot be connected.

### Parameters

- [in] **context** Pointer to the context object
- [in] **enable** true to enable, false to disable
- [out] **error** Pointer to an error object that will be populated if an error occurs.

Referenced by [ob::Context::enableNetDeviceEnumeration\(\)](#).

### ◆ [ob\\_create\\_net\\_device\(\)](#)

```
OB_EXPORT ob_device * ob_create_net_device ( ob_context* context,  
                                         const char* address,  
                                         uint16_t    port,  
                                         ob_error ** error )
```

Create a network device object.

### Parameters

- [in] **context** Pointer to the context object
- [in] **address** IP address of the device
- [in] **port** Port number of the device
- [out] **error** Pointer to an error object that will be populated if an error occurs during device creation

### Returns

Pointer to the created device object

Referenced by [ob::Context::createNetDevice\(\)](#).

◆ **ob\_set\_device\_changed\_callback()**

```
OB_EXPORT void ob_set_device_changed_callback ( ob_context * context,
                                                ob_device_changed_callback callback,
                                                void * user_data,
                                                ob_error ** error )
```

Set a device plug-in callback function.

**Attention**

The added and removed device lists returned through the callback interface need to be released manually

This function supports multiple callbacks. Each call to this function adds a new callback to an internal list.

**Parameters**

[in] **context** Pointer to the context object  
[in] **callback** Pointer to the callback function triggered when a device is plugged or unplugged  
[in] **user\_data** Pointer to user data that can be passed to and retrieved from the callback function  
[out] **error** Pointer to an error object that will be populated if an error occurs during callback function setting

Referenced by **ob::Context::setDeviceChangedCallback()**.

◆ **ob\_enable\_device\_clock\_sync()**

```
OB_EXPORT void ob_enable_device_clock_sync ( ob_context * context,
                                            uint64_t      repeat_interval_msec,
                                            ob_error ** error )
```

Activates device clock synchronization to synchronize the clock of the host and all created devices (if supported).

#### Parameters

[in] <b>context</b>	Pointer to the context object
[in] <b>repeat_interval_msec</b>	The interval for auto-repeated synchronization, in milliseconds. If the value is 0, synchronization is performed only once.
[out] <b>error</b>	Pointer to an error object that will be populated if an error occurs during execution

Referenced by [ob::Context::enableDeviceClockSync\(\)](#).

#### ◆ [ob\\_free\\_idle\\_memory\(\)](#)

```
OB_EXPORT void ob_free_idle_memory ( ob_context * context,
                                       ob_error ** error )
```

Free idle memory from the internal frame memory pool.

#### Parameters

[in] <b>context</b>	Pointer to the context object
[out] <b>error</b>	Pointer to an error object that will be populated if an error occurs during memory freeing

Referenced by [ob::Context::freeIdleMemory\(\)](#).

#### ◆ [ob\\_set\\_uvc\\_backend\\_type\(\)](#)

```
OB_EXPORT void ob_set_uvc_backend_type ( ob_context * context,
                                         ob_uvc_backend_type backend_type,
                                         ob_error ** error )
```

For linux, there are two ways to enable the UVC backend: libuvc and v4l2. This function is used to set the backend type.

It is effective when the new device is created.

### Attention

This interface is only available for Linux.

### Parameters

[in] <b>context</b>	Pointer to the context object
[in] <b>backend_type</b>	The backend type to be used.
[out] <b>error</b>	Pointer to an error object that will be populated if an error occurs during backend type setting

Referenced by [ob::Context::setUvcBackendType\(\)](#).

### ◆ [ob\\_set\\_logger\\_severity\(\)](#)

```
OB_EXPORT void ob_set_logger_severity ( ob_log_severity severity,
                                         ob_error ** error )
```

Set the global log level.

### Attention

This interface setting will affect the output level of all logs (terminal, file, callback)

### Parameters

[in] <b>severity</b>	Log level to set
[out] <b>error</b>	Pointer to an error object that will be populated if an error occurs during log level setting

Referenced by [ob::Context::setLoggerSeverity\(\)](#).

### ◆ [ob\\_set\\_logger\\_to\\_file\(\)](#)

```
OB_EXPORT void ob_set_logger_to_file ( ob_log_severity severity,  
                                     const char *      directory,  
                                     ob_error **       error )
```

Set the log output to a file.

#### Parameters

- [in] **severity** Log level to output to file
- [in] **directory** Path to the log file output directory. If the path is empty, the existing settings will continue to be used (if the existing configuration is also empty, the log will not be output to the file)
- [out] **error** Pointer to an error object that will be populated if an error occurs during log output setting

Referenced by [ob::Context::setLoggerToFile\(\)](#).

#### ◆ [ob\\_set\\_logger\\_to\\_callback\(\)](#)

```
OB_EXPORT void ob_set_logger_to_callback ( ob_log_severity severity,  
                                         ob_log_callback callback,  
                                         void *           user_data,  
                                         ob_error **     error )
```

Set the log callback function.

#### Parameters

- [in] **severity** Log level to set for the callback function
- [in] **callback** Pointer to the callback function
- [in] **user\_data** Pointer to user data that can be passed to and retrieved from the callback function
- [out] **error** Pointer to an error object that will be populated if an error occurs during log callback function setting

Referenced by [ob::Context::setLoggerToCallback\(\)](#).

#### ◆ [ob\\_set\\_logger\\_to\\_console\(\)](#)

```
OB_EXPORT void ob_set_logger_to_console ( ob_log_severity severity,  
                                         ob_error **      error )
```

Set the log output to the console.

### Parameters

- [in] **severity** Log level to output to the console
- [out] **error** Pointer to an error object that will be populated if an error occurs during log output setting

Referenced by [ob::Context::setLoggerToConsole\(\)](#).

### ◆ [ob\\_set\\_extensions\\_directory\(\)](#)

```
OB_EXPORT void ob_set_extensions_directory ( const char * directory,  
                                              ob_error ** error )
```

Set the extensions directory.

The extensions directory is used to search for dynamic libraries that provide additional functionality to the SDK, such as the Frame filters.

### Attention

Should be called before creating the context and pipeline, otherwise the default extensions directory (./extensions) will be used.

### Parameters

- directory** Path to the extensions directory. If the path is empty, extensions path will be set to the current working directory.
- error** Pointer to an error object that will be populated if an error occurs during extensions directory setting

Referenced by [ob::Context::setExtensionsDirectory\(\)](#).

# Context.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
10 #pragma once
11
12 #ifndef __cplusplus
13 extern "C" {
14 #endif
15
16 #include "ObTypes.h"
17
24 OB_EXPORT ob_context *ob_create_context(ob_error **error);
25
33 OB_EXPORT ob_context *ob_create_context_with_config(const char *config_file_path, ob_error **error)
    ;
41 OB_EXPORT void ob_delete_context(ob_context *context, ob_error **error);
42
50 OB_EXPORT ob_device_list *ob_query_device_list(ob_context *context, ob_error **error);
51
64 OB_EXPORT void ob_enable_net_device_enumeration(ob_context *context, bool enable, ob_error **erro
    r);
65
75 OB_EXPORT ob_device *ob_create_net_device(ob_context *context, const char *address, uint16_t port,
    ob_error **error);
76
87 OB_EXPORT void ob_set_device_changed_callback(ob_context *context, ob_device_changed_callback c
    allback, void *user_data, ob_error **error);
88
96 OB_EXPORT void ob_enable_device_clock_sync(ob_context *context, uint64_t repeat_interval_msec, ob
    _error **error);
97
104 OB_EXPORT void ob_free_idle_memory(ob_context *context, ob_error **error);
105
116 OB_EXPORT void ob_set_uvc_backend_type(ob_context *context, ob_uvc_backend_type backend_typ
    e, ob_error **error);
117
126 OB_EXPORT void ob_set_logger_severity(ob_log_severity severity, ob_error **error);
127
136 OB_EXPORT void ob_set_logger_to_file(ob_log_severity severity, const char *directory, ob_error **err
    or);
137
146 OB_EXPORT void ob_set_logger_to_callback(ob_log_severity severity, ob_log_callback callback, void *
    user_data, ob_error **error);
147
154 OB_EXPORT void ob_set_logger_to_console(ob_log_severity severity, ob_error **error);
155
165 OB_EXPORT void ob_set_extensions_directory(const char *directory, ob_error **error);
166
167 // The following interfaces are deprecated and are retained here for compatibility purposes.
168 #define ob_enable_multi_device_sync ob_enable_device_clock_sync
169 #define ob_set_logger_callback ob_set_logger_to_callback
170
171 #ifdef __cplusplus
172 }
173 #endif
```



# Context.hpp File Reference

The SDK context class, which serves as the entry point to the underlying SDK. It is used to query device lists, handle device callbacks, and set the log level. [More...](#)

```
#include "libobsensor/h/Context.h"  
#include "Types.hpp"  
#include "Error.hpp"  
#include <functional>  
#include <memory>
```

[Go to the source code of this file.](#)

## Classes

```
class ob::Context
```

## Namespaces

```
namespace ob
```

## Macros

```
#define enableMultiDeviceSync enableDeviceClockSync
```

## Detailed Description

The SDK context class, which serves as the entry point to the underlying SDK. It is used to query device lists, handle device callbacks, and set the log level.

Definition in file [Context.hpp](#).

## Macro Definition Documentation

### ◆ enableMultiDeviceSync

```
#define enableMultiDeviceSync enableDeviceClockSync
```

Definition at line [249](#) of file [Context.hpp](#).



# Context.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
10 #pragma once
11
12 #include "libobsensor/h/Context.h"
13 #include "Types.hpp"
14 #include "Error.hpp"
15
16 #include <functional>
17 #include <memory>
18
19 namespace ob {
20
21 // forward declarations
22 class Device;
23 class DeviceInfo;
24 class DeviceList;
25
26 class Context {
27 public:
34     typedef std::function<void(std::shared_ptr<DeviceList> removedList, std::shared_ptr<DeviceList> addedList)> DeviceChangedCallback;
35
42     typedef std::function<void(OBLogSeverity severity, const char *logMsg)> LogCallback;
43
44 private:
45     ob_context      *impl_ = nullptr;
46     DeviceChangedCallback deviceChangedCallback_;
47     // static LogCallback logCallback_;
48
49 public:
58     explicit Context(const char *configPath = "") {
59         ob_error *error = nullptr;
60         impl_ = ob_create_context_with_config(configPath, &error);
61         Error::handle(&error);
62     }
63
67     ~Context() noexcept {
68         ob_error *error = nullptr;
69         ob_delete_context(impl_, &error);
70         Error::handle(&error, false);
71     }
72
78     std::shared_ptr<DeviceList> queryDeviceList() const {
79         ob_error *error = nullptr;
80         auto list = ob_query_device_list(impl_, &error);
81         Error::handle(&error);
82         return std::make_shared<DeviceList>(list);
83     }
84
94     void enableNetDeviceEnumeration(bool enable) const {
95         ob_error *error = nullptr;
96         ob_enable_net_device_enumeration(impl_, enable, &error);
97         Error::handle(&error);
98     }
99
107    std::shared_ptr<Device> createNetDevice(const char *address, uint16_t port) const {
```

```

108     ob_error *error = nullptr;
109     auto device = ob_create_net_device(impl_, address, port, &error);
110     Error::handle(&error);
111     return std::make_shared<Device>(device);
112 }
113
120 void setDeviceChangedCallback(DeviceChangedCallback callback) {
121     deviceChangedCallback_ = callback;
122     ob_error *error = nullptr;
123     ob_set_device_changed_callback(impl_, &Context::deviceChangedCallback, this, &error);
124     Error::handle(&error);
125 }
126
132 void enableDeviceClockSync(uint64_t repeatIntervalMsec) const {
133     ob_error *error = nullptr;
134     ob_enable_device_clock_sync(impl_, repeatIntervalMsec, &error);
135     Error::handle(&error);
136 }
137
142 void freeIdleMemory() const {
143     ob_error *error = nullptr;
144     ob_free_idle_memory(impl_, &error);
145     Error::handle(&error);
146 }
147
156 void setUvcBackendType(OBUvcBackendType type) const {
157     ob_error *error = nullptr;
158     ob_set_uvc_backend_type(impl_, type, &error);
159     Error::handle(&error);
160 }
161
168 static void setLoggerSeverity(OBLogSeverity severity) {
169     ob_error *error = nullptr;
170     ob_set_logger_severity(severity, &error);
171     Error::handle(&error);
172 }
173
181 static void setLoggerToFile(OBLogSeverity severity, const char *directory) {
182     ob_error *error = nullptr;
183     ob_set_logger_to_file(severity, directory, &error);
184     Error::handle(&error);
185 }
186
192 static void setLoggerToConsole(OBLogSeverity severity) {
193     ob_error *error = nullptr;
194     ob_set_logger_to_console(severity, &error);
195     Error::handle(&error);
196 }
197
204 static void setLoggerToCallback(OBLogSeverity severity, LogCallback callback) {
205     ob_error *error = nullptr;
206     Context::getLogCallback() = callback;
207     ob_set_logger_to_callback(severity, &Context::logCallback, &Context::getLogCallback(), &error);
208     Error::handle(&error);
209 }
210
219 static void setExtensionsDirectory(const char *directory) {
220     ob_error *error = nullptr;
221     ob_set_extensions_directory(directory, &error);
222     Error::handle(&error);
223 }
224
225 private:
226     static void deviceChangedCallback(ob_device_list *removedList, ob_device_list *addedList, void *use
rData) {

```

```

227     auto ctx = static_cast<Context*>(userData);
228     if(ctx && ctx->deviceChangedCallback_) {
229         auto removed = std::make_shared<DeviceList>(removedList);
230         auto added = std::make_shared<DeviceList>(addedList);
231         ctx->deviceChangedCallback_(removed, added);
232     }
233 }
234
235 static void logCallback(OBLogSeverity severity, const char *logMsg, void *userData) {
236     auto cb = static_cast<LogCallback*>(userData);
237     if(cb) {
238         (*cb)(severity, logMsg);
239     }
240 }
241
242 // Lazy initialization of the logcallback_. The purpose is to initialize logcallback_ in .hpp
243 static LogCallback &getLogCallback() {
244     static LogCallback logCallback_ = nullptr;
245     return logCallback_;
246 }
247
248 // for backward compatibility
249 #define enableMultiDeviceSync enableDeviceClockSync
250 };
251 } // namespace ob

```

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Device.h File Reference

Device-related functions, including operations such as obtaining and creating a device, setting and obtaining device property, and obtaining sensors. [More...](#)

```
#include "ObTypes.h"
#include "Property.h"
#include "MultipleDevices.h"
#include "Advanced.h"
```

Go to the source code of this file.

## Macros

```
#define ob_device_list_device_count ob_device_list_get_count
#define ob_device_list_get_extension_info ob_device_info_get_extension_info
#define ob_device_upgrade ob_device_update_firmware
#define ob_device_upgrade_from_data ob_device_update_firmware_from_data
#define ob_device_get_supported_property ob_device_get_supported_property_item
#define ob_device_state_changed ob_device_set_state_changed_callback
#define ob_device_info_name ob_device_info_get_name
#define ob_device_info_pid ob_device_info_get_pid
#define ob_device_info_vid ob_device_info_get_vid
#define ob_device_info_uid ob_device_info_get_uid
#define ob_device_info_serial_number ob_device_info_get_serial_number
#define ob_device_info_firmware_version ob_device_info_get_firmware_version
#define ob_device_info_connection_type ob_device_info_get_connection_type
#define ob_device_info_ip_address ob_device_info_get_ip_address
#define ob_device_info_hardware_version ob_device_info_get_hardware_version
#define ob_device_info_supported_min_sdk_version ob_device_info_get_supported_min_sdk_version
#define ob_device_info_asicName ob_device_info_get_asicName
#define ob_device_info_device_type ob_device_info_get_device_type
#define ob_device_list_get_device_count ob_device_list_get_count
#define ob_camera_param_list_count ob_camera_param_list_get_count
```

## Functions

```
OB_EXPORT void ob_delete_device (ob_device *device, ob_error **error)
Delete a device.
```

```
OB_EXPORT ob_sensor_list * ob_device_get_sensor_list (const ob_device *device,  
ob_error **error)  
List all sensors.  
OB_EXPORT ob_sensor * ob_device_get_sensor (ob_device *device, ob_sensor_type  
type, ob_error **error)  
Get a device's sensor.  
OB_EXPORT void ob_device_set_int_property (ob_device *device,  
ob_property_id property_id, int32_t value, ob_error **error)  
Set an integer type of device property.  
OB_EXPORT int32_t ob_device_get_int_property (ob_device *device,  
ob_property_id property_id, ob_error **error)  
Get an integer type of device property.  
OB_EXPORT ob_int_property_range ob_device_get_int_property_range (ob_device *device,  
ob_property_id property_id, ob_error **error)  
Get the integer type of device property range.  
OB_EXPORT void ob_device_set_float_property (ob_device *device,  
ob_property_id property_id, float value, ob_error **error)  
Set a float type of device property.  
OB_EXPORT float ob_device_get_float_property (ob_device *device,  
ob_property_id property_id, ob_error **error)  
Get a float type of device property.  
OB_EXPORT ob_float_property_range ob_device_get_float_property_range (ob_device *device,  
ob_property_id property_id, ob_error **error)  
Get the float type of device property range.  
OB_EXPORT void ob_device_set_bool_property (ob_device *device,  
ob_property_id property_id, bool value, ob_error **error)  
Set a boolean type of device property.  
OB_EXPORT bool ob_device_get_bool_property (ob_device *device,  
ob_property_id property_id, ob_error **error)  
Get a boolean type of device property.  
OB_EXPORT ob_bool_property_range ob_device_get_bool_property_range (ob_device *device,  
ob_property_id property_id, ob_error **error)  
Get the boolean type of device property range.  
OB_EXPORT void ob_device_set_structured_data (ob_device *device,  
ob_property_id property_id, const uint8_t *data, uint32_t  
data_size, ob_error **error)  
Set structured data.
```

**OB\_EXPORT** void **ob\_device\_get\_structured\_data** (**ob\_device** \*device,  
**ob\_property\_id** property\_id, uint8\_t \*data, uint32\_t \*data\_size,  
**ob\_error** \*\*error)

Get structured data of a device property.

**OB\_EXPORT** void **ob\_device\_get\_raw\_data** (**ob\_device** \*device, **ob\_property\_id**  
property\_id, **ob\_get\_data\_callback** cb, void \*user\_data,  
**ob\_error** \*\*error)

Get raw data of a device property.

**OB\_EXPORT** void **ob\_device\_write\_customer\_data** (**ob\_device** \*device, const  
void \*data, uint32\_t data\_size, **ob\_error** \*\*error)

Set customer data.

**OB\_EXPORT** void **ob\_device\_read\_customer\_data** (**ob\_device** \*device, void  
\*data, uint32\_t \*data\_size, **ob\_error** \*\*error)

Get customer data of a device property.

**OB\_EXPORT** uint32\_t **ob\_device\_get\_supported\_property\_count** (const **ob\_device**  
\*device, **ob\_error** \*\*error)

Get the number of properties supported by the device.

**OB\_EXPORT** **ob\_property\_item** **ob\_device\_get\_supported\_property\_item** (const **ob\_device**  
\*device, uint32\_t index, **ob\_error** \*\*error)

Get the type of property supported by the device.

**OB\_EXPORT** bool **ob\_device\_is\_property\_supported** (const **ob\_device** \*device,  
**ob\_property\_id** property\_id, **ob\_permission\_type** permission,  
**ob\_error** \*\*error)

Check if a device property permission is supported.

**OB\_EXPORT** bool **ob\_device\_is\_global\_timestamp\_supported** (const **ob\_device**  
\*device, **ob\_error** \*\*error)

Check if the device supports global timestamp.

**OB\_EXPORT** void **ob\_device\_enable\_global\_timestamp** (**ob\_device** \*device,  
bool enable, **ob\_error** \*\*error)

Enable or disable global timestamp.

**OB\_EXPORT** void **ob\_device\_update\_firmware** (**ob\_device** \*device, const char  
\*path, **ob\_device\_fw\_update\_callback** callback, bool async,  
void \*user\_data, **ob\_error** \*\*error)

Update the device firmware.

**OB\_EXPORT** void **ob\_device\_update\_firmware\_from\_data** (**ob\_device** \*device,  
const uint8\_t \*data, uint32\_t data\_size,  
**ob\_device\_fw\_update\_callback** callback, bool async, void  
\*user\_data, **ob\_error** \*\*error)

Update the device firmware from data.

```
OB_EXPORT void ob_device_update_optional_depth_presets (ob_device
    *device, const char file_path_list[][OB_PATH_MAX], uint8_t
    path_count, ob_device_fw_update_callback callback, void
    *user_data, ob_error **error)
    Update the device optional depth presets.

OB_EXPORT void ob_device_reboot (ob_device *device, ob_error **error)
    Device reboot.

OB_EXPORT ob_device_state ob_device_get_device_state (const ob_device *device,
    ob_error **error)
    Get the current device status.

OB_EXPORT void ob_device_set_state_changed_callback (ob_device *device,
    ob_device_state_callback callback, void *user_data, ob_error
    **error)
    Set the device state changed callback.

OB_EXPORT void ob_device_enable_heartbeat (ob_device *device, bool enable,
    ob_error **error)
    Enable or disable the device heartbeat.

OB_EXPORT void ob_device_send_and_receive_data (ob_device *device, const
    uint8_t *send_data, uint32_t send_data_size, uint8_t
    *receive_data, uint32_t *receive_data_size, ob_error **error)
    Send data to the device and receive data from the device.

OB_EXPORT ob_device_info * ob_device_get_device_info (const ob_device *device,
    ob_error **error)
    Get device information.

OB_EXPORT void ob_delete_device_info (ob_device_info *info, ob_error
    **error)
    Delete device information.

OB_EXPORT const char * ob_device_info_get_name (const ob_device_info *info,
    ob_error **error)
    Get device name.

OB_EXPORT int ob_device_info_get_pid (const ob_device_info *info,
    ob_error **error)
    Get device pid.

OB_EXPORT int ob_device_info_get_vid (const ob_device_info *info,
    ob_error **error)
    Get device vid.

OB_EXPORT const char * ob_device_info_get_uid (const ob_device_info *info,
    ob_error **error)
    Get device uid.
```

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_serial\_number** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get device serial number.

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_firmware\_version** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the firmware version number.

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_connection\_type** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the device connection type.

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_ip\_address** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the device IP address.

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_hardware\_version** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the hardware version number.

**OB\_EXPORT** bool **ob\_device\_is\_extension\_info\_exist** (const **ob\_device** \*device, const char \*info\_key, **ob\_error** \*\*error)

Check if the device extension information exists.

**OB\_EXPORT** const char \* **ob\_device\_get\_extension\_info** (const **ob\_device** \*device, const char \*info\_key, **ob\_error** \*\*error)

Get the device extension information.

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_supported\_min\_sdk\_version** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the minimum SDK version number supported by the device.

**OB\_EXPORT** const char \* **ob\_device\_info\_get\_asicName** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the chip name.

**OB\_EXPORT** **ob\_device\_type** **ob\_device\_info\_get\_device\_type** (const **ob\_device\_info** \*info, **ob\_error** \*\*error)

Get the device type.

**OB\_EXPORT** void **ob\_delete\_device\_list** (**ob\_device\_list** \*list, **ob\_error** \*\*error)

Delete a device list.

**OB\_EXPORT** uint32\_t **ob\_device\_list\_get\_count** (const **ob\_device\_list** \*list, **ob\_error** \*\*error)

Get the number of devices.

**OB\_EXPORT** const char \* **ob\_device\_list\_get\_device\_name** (const **ob\_device\_list** \*list, uint32\_t index, **ob\_error** \*\*error)

Get device name.

**OB\_EXPORT** int **ob\_device\_list\_get\_device\_pid**(const **ob\_device\_list** \*list,  
                  uint32\_t index, **ob\_error** \*\*error)  
                Get the pid of the specified device.

**OB\_EXPORT** int **ob\_device\_list\_get\_device\_vid**(const **ob\_device\_list** \*list,  
                  uint32\_t index, **ob\_error** \*\*error)  
                Get the vid of the specified device.

**OB\_EXPORT** const char \* **ob\_device\_list\_get\_device\_uid**(const **ob\_device\_list** \*list,  
                  uint32\_t index, **ob\_error** \*\*error)  
                Get the uid of the specified device.

**OB\_EXPORT** const char \* **ob\_device\_list\_get\_device\_serial\_number**(const  
                         **ob\_device\_list** \*list, uint32\_t index, **ob\_error** \*\*error)  
                Get the serial number of the specified device.

**OB\_EXPORT** const char \* **ob\_device\_list\_get\_device\_connection\_type**(const  
                         **ob\_device\_list** \*list, uint32\_t index, **ob\_error** \*\*error)  
                Get device connection type.

**OB\_EXPORT** const char \* **ob\_device\_list\_get\_device\_ip\_address**(const **ob\_device\_list**  
                         \*list, uint32\_t index, **ob\_error** \*\*error)  
                Get device ip address.

**OB\_EXPORT** const char \* **ob\_device\_list\_get\_device\_local\_mac**(const **ob\_device\_list**  
                         \*list, uint32\_t index, **ob\_error** \*\*error)  
                Get device local mac address.

**OB\_EXPORT** **ob\_device** \* **ob\_device\_list\_get\_device**(const **ob\_device\_list** \*list,  
                  uint32\_t index, **ob\_error** \*\*error)  
                Create a device.

**OB\_EXPORT** **ob\_device** \* **ob\_device\_list\_get\_device\_by\_serial\_number**(const  
                         **ob\_device\_list** \*list, const char \*serial\_number, **ob\_error**  
                         \*\*error)  
                Create a device.

**OB\_EXPORT** **ob\_device** \* **ob\_device\_list\_get\_device\_by\_uid**(const **ob\_device\_list** \*list,  
                  const char \*uid, **ob\_error** \*\*error)  
                Create device by uid.

**OB\_EXPORT** **ob\_camera\_param\_list** \* **ob\_device\_get\_calibration\_camera\_param\_list**(**ob\_device**  
                         \*device, **ob\_error** \*\*error)  
                Get the original parameter list of camera calibration saved on  
                the device.

**OB\_EXPORT** uint32\_t **ob\_camera\_param\_list\_get\_count**(**ob\_camera\_param\_list**  
                         \*param\_list, **ob\_error** \*\*error)  
                Get the number of camera parameter lists.

```
OB_EXPORT ob_camera_param ob_camera_param_list_get_param (ob_camera_param_list  
*param_list, uint32_t index, ob_error **error)
```

Get camera parameters from the camera parameter list.

```
OB_EXPORT void ob_delete_camera_param_list (ob_camera_param_list  
*param_list, ob_error **error)
```

Delete the camera parameter list.

## Detailed Description

Device-related functions, including operations such as obtaining and creating a device, setting and obtaining device property, and obtaining sensors.

Definition in file **Device.h**.

## Macro Definition Documentation

### ◆ ob\_device\_list\_device\_count

```
#define ob_device_list_device_count ob_device_list_get_count
```

Definition at line **667** of file **Device.h**.

### ◆ ob\_device\_list\_get\_extension\_info

```
#define ob_device_list_get_extension_info ob_device_info_get_extension_info
```

Definition at line **668** of file **Device.h**.

### ◆ ob\_device\_upgrade

```
#define ob_device_upgrade ob_device_update_firmware
```

Definition at line **669** of file **Device.h**.

### ◆ ob\_device\_upgrade\_from\_data

```
#define ob_device_upgrade_from_data ob_device_update_firmware_from_data
```

Definition at line **670** of file **Device.h**.

◆ **ob\_device\_get\_supported\_property**

```
#define ob_device_get_supported_property ob_device_get_supported_property_item
```

Definition at line **671** of file **Device.h**.

◆ **ob\_device\_state\_changed**

```
#define ob_device_state_changed ob_device_set_state_changed_callback
```

Definition at line **672** of file **Device.h**.

◆ **ob\_device\_info\_name**

```
#define ob_device_info_name ob_device_info_get_name
```

Definition at line **673** of file **Device.h**.

◆ **ob\_device\_info\_pid**

```
#define ob_device_info_pid ob_device_info_get_pid
```

Definition at line **674** of file **Device.h**.

◆ **ob\_device\_info\_vid**

```
#define ob_device_info_vid ob_device_info_get_vid
```

Definition at line **675** of file **Device.h**.

◆ **ob\_device\_info\_uid**

```
#define ob_device_info_uid ob_device_info_get_uid
```

Definition at line **676** of file **Device.h**.

◆ **ob\_device\_info\_serial\_number**

```
#define ob_device_info_serial_number ob_device_info_get_serial_number
```

Definition at line **677** of file **Device.h**.

◆ **ob\_device\_info\_firmware\_version**

```
#define ob_device_info_firmware_version ob_device_info_get_firmware_version
```

Definition at line **678** of file **Device.h**.

◆ **ob\_device\_info\_connection\_type**

```
#define ob_device_info_connection_type ob_device_info_get_connection_type
```

Definition at line **679** of file **Device.h**.

◆ **ob\_device\_info\_ip\_address**

```
#define ob_device_info_ip_address ob_device_info_get_ip_address
```

Definition at line **680** of file **Device.h**.

◆ **ob\_device\_info\_hardware\_version**

```
#define ob_device_info_hardware_version ob_device_info_get_hardware_version
```

Definition at line **681** of file **Device.h**.

◆ **ob\_device\_info\_supported\_min\_sdk\_version**

```
#define ob_device_info_supported_min_sdk_version ob_device_info_get_supported_min_sdk_version
```

Definition at line **682** of file **Device.h**.

◆ **ob\_device\_info\_asicName**

```
#define ob_device_info_asicName ob_device_info_get_asicName
```

Definition at line **683** of file **Device.h**.

◆ **ob\_device\_info\_device\_type**

```
#define ob_device_info_device_type ob_device_info_get_device_type
```

Definition at line **684** of file **Device.h**.

◆ **ob\_device\_list\_get\_device\_count**

```
#define ob_device_list_get_device_count ob_device_list_get_count
```

Definition at line **685** of file **Device.h**.

◆ **ob\_camera\_param\_list\_count**

```
#define ob_camera_param_list_count ob_camera_param_list_get_count
```

Definition at line **686** of file **Device.h**.

## Function Documentation

◆ **ob\_delete\_device()**

```
OB_EXPORT void ob_delete_device ( ob_device * device,  
                                ob_error ** error )
```

Delete a device.

#### Parameters

- [in] **device** The device to be deleted.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Device::operator=()**, and **ob::Device::~Device()**.

#### ◆ **ob\_device\_get\_sensor\_list()**

```
OB_EXPORT ob_sensor_list * ob_device_get_sensor_list ( const ob_device * device,  
                                              ob_error ** error )
```

List all sensors.

#### Parameters

- [in] **device** The device object.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_sensor\_list\*** The list of all sensors.

Referenced by **ob::Device::getSensorList()**.

#### ◆ **ob\_device\_get\_sensor()**

```
OB_EXPORT ob_sensor* ob_device_get_sensor ( ob_device * device,  
                                              ob_sensor_type type,  
                                              ob_error ** error )
```

Get a device's sensor.

#### Parameters

- [in] **device** The device object.
- [in] **type** The type of sensor to get.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

ob\_sensor\* The acquired sensor.

Referenced by [ob::Device::getSensor\(\)](#).

#### ◆ [ob\\_device\\_set\\_int\\_property\(\)](#)

```
OB_EXPORT void ob_device_set_int_property ( ob_device * device,  
                                              ob_property_id property_id,  
                                              int32_t value,  
                                              ob_error ** error )
```

Set an integer type of device property.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The ID of the property to be set.
- [in] **value** The property value to be set.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::setIntProperty\(\)](#).

#### ◆ [ob\\_device\\_get\\_int\\_property\(\)](#)

```
OB_EXPORT int32_t ob_device_get_int_property ( ob_device * device,  
                                              ob_property_id property_id,  
                                              ob_error ** error )
```

Get an integer type of device property.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The property ID.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

int32\_t The property value.

Referenced by [ob::Device::getIntProperty\(\)](#).

#### ◆ [ob\\_device\\_get\\_int\\_property\\_range\(\)](#)

```
OB_EXPORT ob_int_property_range ob_device_get_int_property_range ( ob_device * device,  
                                                               ob_property_id property_id,  
                                                               ob_error ** error )
```

Get the integer type of device property range.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The property id.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The property range.

Referenced by [ob::Device::getIntPropertyRange\(\)](#).

#### ◆ [ob\\_device\\_set\\_float\\_property\(\)](#)

```
OB_EXPORT void ob_device_set_float_property ( ob_device * device,  
                                              ob_property_id property_id,  
                                              float value,  
                                              ob_error ** error )
```

Set a float type of device property.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The ID of the property to be set.
- [in] **value** The property value to be set.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::setFloatProperty\(\)](#).

#### ◆ [ob\\_device\\_get\\_float\\_property\(\)](#)

```
OB_EXPORT float ob_device_get_float_property ( ob_device * device,  
                                              ob_property_id property_id,  
                                              ob_error ** error )
```

Get a float type of device property.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The property ID.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

float The property value.

Referenced by [ob::Device::getFloatProperty\(\)](#).

#### ◆ [ob\\_device\\_get\\_float\\_property\\_range\(\)](#)

```
OB_EXPORT ob_float_property_range ob_device_get_float_property_range ( ob_device * device,  
                                         ob_property_id property_id,  
                                         ob_error ** error )
```

Get the float type of device property range.

#### Parameters

- [in] **device**      The device object.
- [in] **property\_id** The property id.
- [out] **error**        Pointer to an error object that will be set if an error occurs.

#### Returns

The property range.

Referenced by [ob::Device::getFloatPropertyRange\(\)](#).

#### ◆ [ob\\_device\\_set\\_bool\\_property\(\)](#)

```
OB_EXPORT void ob_device_set_bool_property ( ob_device * device,  
                                         ob_property_id property_id,  
                                         bool                value,  
                                         ob_error ** error )
```

Set a boolean type of device property.

#### Parameters

- [in] **device**      The device object.
- [in] **property\_id** The ID of the property to be set.
- [in] **value**        The property value to be set.
- [out] **error**        Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::setBoolProperty\(\)](#).

#### ◆ [ob\\_device\\_get\\_bool\\_property\(\)](#)

```
OB_EXPORT bool ob_device_get_bool_property ( ob_device * device,  
                                              ob_property_id property_id,  
                                              ob_error ** error )
```

Get a boolean type of device property.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The property ID.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**bool** The property value.

Referenced by [ob::Device::getBoolProperty\(\)](#).

### ◆ [ob\\_device\\_get\\_bool\\_property\\_range\(\)](#)

```
OB_EXPORT ob_bool_property_range ob_device_get_bool_property_range ( ob_device * device,  
                                                               ob_property_id property_id,  
                                                               ob_error ** error )
```

Get the boolean type of device property range.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The property id.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The property range.

Referenced by [ob::Device::getBoolPropertyRange\(\)](#).

### ◆ [ob\\_device\\_set\\_structured\\_data\(\)](#)

```
OB_EXPORT void ob_device_set_structured_data ( ob_device * device,  
                                              ob_property_id property_id,  
                                              const uint8_t * data,  
                                              uint32_t data_size,  
                                              ob_error ** error )
```

Set structured data.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The ID of the property to be set.
- [in] **data** The property data to be set.
- [in] **data\_size** The size of the property to be set.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::setStructuredData\(\)](#).

#### ◆ [ob\\_device\\_get\\_structured\\_data\(\)](#)

```
OB_EXPORT void ob_device_get_structured_data ( ob_device * device,  
                                              ob_property_id property_id,  
                                              uint8_t * data,  
                                              uint32_t * data_size,  
                                              ob_error ** error )
```

Get structured data of a device property.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The ID of the property.
- [out] **data** The obtained property data.
- [out] **data\_size** The size of the obtained property data.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::getStructuredData\(\)](#).

#### ◆ [ob\\_device\\_get\\_raw\\_data\(\)](#)

```
OB_EXPORT void ob_device_get_raw_data ( ob_device * device,  
                                         ob_property_id property_id,  
                                         ob_get_data_callback cb,  
                                         void * user_data,  
                                         ob_error ** error )
```

Get raw data of a device property.

### Parameters

- [in] **device** The device object.
- [in] **property\_id** The ID of the property.
- [out] **cb** The get data callback.
- [out] **user\_data** User-defined data that will be returned in the callback.
- [out] **error** Pointer to an error object that will be set if an error occurs.

## ◆ **ob\_device\_write\_customer\_data()**

```
OB_EXPORT void ob_device_write_customer_data ( ob_device * device,  
                                              const void * data,  
                                              uint32_t data_size,  
                                              ob_error ** error )
```

Set customer data.

### Parameters

- [in] **device** The device object.
- [in] **data** The property data to be set.
- [in] **data\_size** The size of the property to be set, the maximum length cannot exceed 65532 bytes.
- [out] **error** Log error messages.

Referenced by **ob::Device::writeCustomerData()**.

## ◆ **ob\_device\_read\_customer\_data()**

```
OB_EXPORT void ob_device_read_customer_data ( ob_device * device,  
                                              void *      data,  
                                              uint32_t *   data_size,  
                                              ob_error ** error )
```

Get customer data of a device property.

#### Parameters

- [in] **device** The device object.
- [out] **data** The obtained property data.
- [out] **data\_size** The size of the obtained property data.
- [out] **error** Log error messages.

Referenced by [ob::Device::readCustomerData\(\)](#).

#### ◆ [ob\\_device\\_get\\_supported\\_property\\_count\(\)](#)

```
OB_EXPORT uint32_t ob_device_get_supported_property_count ( const ob_device * device,  
                                                               ob_error **      error )
```

Get the number of properties supported by the device.

#### Parameters

- [in] **device** The device object.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The number of properties supported by the device.

Referenced by [ob::Device::getSupportedPropertyCount\(\)](#).

#### ◆ [ob\\_device\\_get\\_supported\\_property\\_item\(\)](#)

```
OB_EXPORT ob_property_item ob_device_get_supported_property_item( const ob_device * device,  
                                  uint32_t              index,  
                                  ob_error **          error )
```

Get the type of property supported by the device.

#### Parameters

- [in] **device** The device object.
- [in] **index** The property index.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The type of property supported by the device.

Referenced by [ob::Device::getSupportedProperty\(\)](#).

### ◆ [ob\\_device\\_is\\_property\\_supported\(\)](#)

```
OB_EXPORT bool ob_device_is_property_supported( const ob_device *    device,  
                                  ob_property_id       property_id,  
                                  ob_permission_type permission,  
                                  ob_error **          error )
```

Check if a device property permission is supported.

#### Parameters

- [in] **device** The device object.
- [in] **property\_id** The property id.
- [in] **permission** The type of permission that needs to be interpreted.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

Whether the property permission is supported.

Referenced by [ob::Device::isPropertySupported\(\)](#).

### ◆ [ob\\_device\\_is\\_global\\_timestamp\\_supported\(\)](#)

Check if the device supports global timestamp.

## Parameters

[in] **device** The device object.

**[out] error** Pointer to an error object that will be set if an error occurs.

## Returns

bool Whether the device supports global timestamp.

Referenced by **ob::Device::isGlobalTimestampSupported()**.

#### ◆ ob\_device\_enable\_global\_timestamp()

Enable or disable global timestamp.

## Parameters

**device** The device object.

**enable** Whether to enable global timestamp.

**error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::enableGlobalTimestamp\(\)](#).

#### ◆ ob\_device\_update\_firmware()

```
OB_EXPORT void ob_device_update_firmware ( ob_device * device,  
                                         const char * path,  
                                         ob_device_fw_update_callback callback,  
                                         bool async,  
                                         void * user_data,  
                                         ob_error ** error )
```

Update the device firmware.

### Parameters

[in] **device** The device object.

[in] **path** The firmware path.

[in] **callback** The firmware upgrade progress callback.

[in] **async** Whether to execute asynchronously.

[in] **user\_data** User-defined data that will be returned in the callback.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::updateFirmware\(\)](#).

◆ [ob\\_device\\_update\\_firmware\\_from\\_data\(\)](#)

```
OB_EXPORT void ob_device_update_firmware_from_data ( ob_device * device,  
                                                 const uint8_t * data,  
                                                 uint32_t data_size,  
                                                 ob_device_fw_update_callback callback,  
                                                 bool async,  
                                                 void * user_data,  
                                                 ob_error ** error )
```

Update the device firmware from data.

## Parameters

- [in] **device** The device object.
- [in] **data** The firmware file data.
- [in] **data\_size** The firmware file size.
- [in] **callback** The firmware upgrade progress callback.
- [in] **async** Whether to execute asynchronously.
- [in] **user\_data** User-defined data that will be returned in the callback.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::updateFirmwareFromData\(\)](#).

- ◆ [ob\\_device\\_update\\_optional\\_depth\\_presets\(\)](#)

```
OB_EXPORT void
ob_device_update_optional_depth_presets ( ob_device * device,
                                         const char file_path_list[][OB_PATH_MAX],
                                         uint8_t path_count,
                                         ob_device_fw_update_callback callback,
                                         void * user_data,
                                         ob_error ** error )
```

Update the device optional depth presets.

### Parameters

- [in] **device** The device object.
- [in] **file\_path\_list** A list(2D array) of preset file paths, each up to OB\_PATH\_MAX characters.
- [in] **path\_count** The number of the preset file paths.
- [in] **callback** The preset upgrade progress callback.
- [in] **user\_data** User-defined data that will be returned in the callback.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::updateOptionalDepthPresets\(\)](#).

### ◆ ob\_device\_reboot()

```
OB_EXPORT void ob_device_reboot ( ob_device * device,
                                  ob_error ** error )
```

Device reboot.

### Attention

The device will be disconnected and reconnected. After the device is disconnected, the interface access to the device handle may be abnormal. Please use the ob\_delete\_device interface to delete the handle directly. After the device is reconnected, it can be obtained again.

### Parameters

- [in] **device** Device object
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::reboot\(\)](#).

### ◆ ob\_device\_get\_device\_state()

```
OB_EXPORT ob_device_state ob_device_get_device_state ( const ob_device * device,  
                                              ob_error **          error )
```

Get the current device status.

### Parameters

- [in] **device** The device object.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_device\_state** The device state information.

Referenced by [ob::Device::getDeviceState\(\)](#).

### ◆ [ob\\_device\\_set\\_state\\_changed\\_callback\(\)](#)

```
OB_EXPORT void ob_device_set_state_changed_callback ( ob_device *           device,  
                                                 ob_device_state_callback callback,  
                                                 void *                 user_data,  
                                                 ob_error **            error )
```

Set the device state changed callback.

### Parameters

- [in] **device** The device object.
- [in] **callback** The callback function to be called when the device status changes.
- [in] **user\_data** User-defined data that will be returned in the callback.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::setDeviceStateChangedCallback\(\)](#).

### ◆ [ob\\_device\\_enable\\_heartbeat\(\)](#)

```
OB_EXPORT void ob_device_enable_heartbeat ( ob_device * device,  
                                         bool          enable,  
                                         ob_error ** error )
```

Enable or disable the device heartbeat.

After enable the device heartbeat, the sdk will start a thread to send heartbeat signal to the device error every 3 seconds.

### Attention

If the device does not receive the heartbeat signal for a long time, it will be disconnected and rebooted.

### Parameters

[in] **device** The device object.

[in] **enable** Whether to enable the device heartbeat.

**error** Pointer to an error object that will be set if an error occurs.

Referenced by [\*\*ob::Device::enableHeartbeat\(\)\*\*](#).

- ◆ [ob\\_device\\_send\\_and\\_receive\\_data\(\)](#)

```
OB_EXPORT void ob_device_send_and_receive_data ( ob_device * device,
                                                const uint8_t * send_data,
                                                uint32_t send_data_size,
                                                uint8_t * receive_data,
                                                uint32_t * receive_data_size,
                                                ob_error ** error )
```

Send data to the device and receive data from the device.

This is a factory and debug function, which can be used to send and receive data from the device. The data format is secret and belongs to the device vendor.

### Parameters

[in]	<b>device</b>	The device object.
[in]	<b>send_data</b>	The data to be sent to the device.
[in]	<b>send_data_size</b>	The size of the data to be sent to the device.
[out]	<b>receive_data</b>	The data received from the device.
[in,out]	<b>receive_data_size</b>	Pass in the expected size of the receive data, and return the actual size of the received data.
	<b>error</b>	Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::sendAndReceiveData\(\)](#).

### ◆ [ob\\_device\\_get\\_device\\_info\(\)](#)

```
OB_EXPORT ob_device_info * ob_device_get_device_info ( const ob_device * device,
                                                       ob_error ** error )
```

Get device information.

### Parameters

[in]	<b>device</b>	The device to obtain information from.
[out]	<b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_device\_info**\* The device information.

Referenced by [ob::Device::getDeviceInfo\(\)](#).

### ◆ [ob\\_delete\\_device\\_info\(\)](#)

```
OB_EXPORT void ob_delete_device_info ( ob_device_info * info,
                                         ob_error **      error )
```

Delete device information.

### Parameters

- [in] **info** The device information to be deleted.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::DeviceInfo::~DeviceInfo()**.

- ◆ **ob\_device\_info\_get\_name()**

```
OB_EXPORT const char * ob_device_info_get_name ( const ob_device_info * info,
                                                 ob_error **          error )
```

Get device name.

### Parameters

- [in] **info** Device Information
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

const char\* return the device name

Referenced by **ob::DeviceInfo::getName()**.

- ◆ **ob\_device\_info\_get\_pid()**

```
OB_EXPORT int ob_device_info_get_pid ( const ob_device_info * info,  
                                         ob_error **           error )
```

Get device pid.

### Parameters

[in] **info** Device Information  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

int return the device pid

Referenced by [ob::DeviceInfo::getPid\(\)](#).

### ◆ [ob\\_device\\_info\\_get\\_vid\(\)](#)

```
OB_EXPORT int ob_device_info_get_vid ( const ob_device_info * info,  
                                         ob_error **           error )
```

Get device vid.

### Parameters

[in] **info** Device Information  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

int return device vid

Referenced by [ob::DeviceInfo::getVid\(\)](#).

### ◆ [ob\\_device\\_info\\_get\\_uid\(\)](#)

```
OB_EXPORT const char* ob_device_info_get_uid ( const ob_device_info* info,  
                                              ob_error **           error )
```

Get device uid.

#### Parameters

[in] **info** Device Information  
[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* return device uid

Referenced by [ob::DeviceInfo::getUid\(\)](#).

#### ◆ [ob\\_device\\_info\\_get\\_serial\\_number\(\)](#)

```
OB_EXPORT const char* ob_device_info_get_serial_number ( const ob_device_info* info,  
                                                       ob_error **           error )
```

Get device serial number.

#### Parameters

[in] **info** Device Information  
[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* return device serial number

Referenced by [ob::DeviceInfo::getSerialNumber\(\)](#).

#### ◆ [ob\\_device\\_info\\_get\\_firmware\\_version\(\)](#)

Get the firmware version number.

## Parameters

[in] **info** Device Information

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

int return the firmware version number

Referenced by [ob::DeviceInfo::getFirmwareVersion\(\)](#).

- #### ◆ ob\_device\_info\_get\_connection\_type()

Get the device connection type.

## Parameters

[in] **info** Device Information

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

const char\* The connection type, currently supports: "USB", "USB1.0", "USB1.1", "USB2.0", "USB2.1", "USB3.0", "USB3.1", "USB3.2", "Ethernet"

Referenced by [ob::DeviceInfo::getConnectionType\(\)](#).

- #### ◆ ob\_device\_info\_get\_ip\_address()

```
OB_EXPORT const char* ob_device_info_get_ip_address ( const ob_device_info* info,
                                                       ob_error **           error )
```

Get the device IP address.

### Attention

Only valid for network devices, otherwise it will return "0.0.0.0"

### Parameters

**info** Device Information

**error** Pointer to an error object that will be set if an error occurs.

### Returns

const char\* The IP address, such as "192.168.1.10"

Referenced by [ob::DeviceInfo::getIpAddress\(\)](#).

### ◆ [ob\\_device\\_info\\_get\\_hardware\\_version\(\)](#)

```
OB_EXPORT const char* ob_device_info_get_hardware_version ( const ob_device_info* info,
                                                               ob_error **           error )
```

Get the hardware version number.

### Parameters

[in] **info** Device Information

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

const char\* The hardware version number

Referenced by [ob::DeviceInfo::getHardwareVersion\(\)](#).

### ◆ [ob\\_device\\_is\\_extension\\_info\\_exist\(\)](#)

```
OB_EXPORT bool ob_device_is_extension_info_exist ( const ob_device * device,  
                                              const char *      info_key,  
                                              ob_error **      error )
```

Check if the device extension information exists.

#### Parameters

- device** The device object.
- info\_key** The key of the device extension information.
- error** Pointer to an error object that will be set if an error occurs.

#### Returns

bool Whether the device extension information exists.

Referenced by [ob::Device::isExtensionInfoExist\(\)](#).

### ◆ [ob\\_device\\_get\\_extension\\_info\(\)](#)

```
OB_EXPORT const char * ob_device_get_extension_info ( const ob_device * device,  
                                              const char *      info_key,  
                                              ob_error **      error )
```

Get the device extension information.

Extension information is a set of key-value pair of string, user can get the information by the key.

#### Parameters

- [in] **device** The device object.
- [in] **info\_key** The key of the device extension information.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* The device extension information

Referenced by [ob::Device::getExtensionInfo\(\)](#).

### ◆ [ob\\_device\\_info\\_get\\_supported\\_min\\_sdk\\_version\(\)](#)

```
OB_EXPORT const char* ob_device_info_get_supported_min_sdk_version ( const ob_device_info * info,
ob_error ** error )
```

Get the minimum SDK version number supported by the device.

#### Parameters

- [in] **info** Device Information
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* The minimum SDK version number supported by the device

Referenced by [ob::DeviceInfo::getSupportedMinSdkVersion\(\)](#).

#### ◆ [ob\\_device\\_info\\_get\\_asicName\(\)](#)

```
OB_EXPORT const char* ob_device_info_get_asicName ( const ob_device_info * info,
ob_error ** error )
```

Get the chip name.

#### Parameters

- [in] **info** Device Information
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* The ASIC name

Referenced by [ob::DeviceInfo::getAsicName\(\)](#).

#### ◆ [ob\\_device\\_info\\_get\\_device\\_type\(\)](#)

Get the device type.

## Parameters

[in] **info** Device Information

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob device type** The device type

Referenced by **ob::DeviceInfo::getDeviceType()**.

- #### ◆ ob\_delete\_device\_list()

Delete a device list.

## Parameters

[in] **list** The device list object to be deleted.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::DeviceList::~DeviceList()**.

- #### ◆ ob\_device\_list\_get\_count()

```
OB_EXPORT uint32_t ob_device_list_get_count ( const ob_device_list * list,  
                                              ob_error **          error )
```

Get the number of devices.

#### Parameters

- [in] **list** Device list object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

uint32\_t return the number of devices

Referenced by [ob::DeviceList::getCount\(\)](#).

#### ◆ **ob\_device\_list\_get\_device\_name()**

```
OB_EXPORT const char * ob_device_list_get_device_name ( const ob_device_list * list,  
                                                       uint32_t           index,  
                                                       ob_error **        error )
```

Get device name.

#### Parameters

- [in] **list** Device list object
- [in] **index** Device index
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* return device name

Referenced by [ob::DeviceList::getName\(\)](#).

#### ◆ **ob\_device\_list\_get\_device\_pid()**

```
OB_EXPORT int ob_device_list_get_device_pid ( const ob_device_list* list,
                                              uint32_t           index,
                                              ob_error **       error )
```

Get the pid of the specified device.

#### Parameters

- [in] **list** Device list object
- [in] **index** Device index
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

int return the device pid

Referenced by [ob::DeviceList::getPid\(\)](#).

### ◆ [ob\\_device\\_list\\_get\\_device\\_vid\(\)](#)

```
OB_EXPORT int ob_device_list_get_device_vid ( const ob_device_list* list,
                                               uint32_t           index,
                                               ob_error **       error )
```

Get the vid of the specified device.

#### Parameters

- [in] **list** Device list object
- [in] **index** Device index
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

int return device vid

Referenced by [ob::DeviceList::getVid\(\)](#).

### ◆ [ob\\_device\\_list\\_get\\_device\\_uid\(\)](#)

```
OB_EXPORT const char* ob_device_list_get_device_uid ( const ob_device_list* list,
                                                       uint32_t index,
                                                       ob_error ** error )
```

Get the uid of the specified device.

#### Parameters

- [in] **list** Device list object
- [in] **index** Device index
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* return the device uid

Referenced by [ob::DeviceList::getUid\(\)](#).

### ◆ [ob\\_device\\_list\\_get\\_device\\_serial\\_number\(\)](#)

```
OB_EXPORT const char* ob_device_list_get_device_serial_number ( const ob_device_list* list,
                                                               uint32_t index,
                                                               ob_error ** error )
```

Get the serial number of the specified device.

#### Parameters

- [in] **list** Device list object.
- [in] **index** Device index.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* The device UID.

Referenced by [ob::DeviceList::getSerialNumber\(\)](#).

### ◆ [ob\\_device\\_list\\_get\\_device\\_connection\\_type\(\)](#)

```
OB_EXPORT const char* ob_device_list_get_device_connection_type ( const ob_device_list* list,
                                                               uint32_t index,
                                                               ob_error ** error )
```

Get device connection type.

#### Parameters

- [in] **list** Device list object
- [in] **index** Device index
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* returns the device connection type, currently supports: "USB", "USB1.0", "USB1.1", "USB2.0", "USB2.1", "USB3.0", "USB3.1", "USB3.2", "Ethernet"

Referenced by [ob::DeviceList::getConnectionType\(\)](#).

### ◆ [ob\\_device\\_list\\_get\\_device\\_ip\\_address\(\)](#)

```
OB_EXPORT const char* ob_device_list_get_device_ip_address ( const ob_device_list* list,
                                                               uint32_t index,
                                                               ob_error ** error )
```

Get device ip address.

#### Attention

Only valid for network devices, otherwise it will return "0.0.0.0".

#### Parameters

- list** Device list object
- index** Device index
- error** Pointer to an error object that will be set if an error occurs.

#### Returns

const char\* returns the device ip address, such as "192.168.1.10"

Referenced by [ob::DeviceList::getIpAddress\(\)](#).

### ◆ [ob\\_device\\_list\\_get\\_device\\_local\\_mac\(\)](#)

```
OB_EXPORT const char* ob_device_list_get_device_local_mac ( const ob_device_list* list,
                                                               uint32_t index,
                                                               ob_error ** error )
```

Get device local mac address.

### Attention

Only valid for network devices, otherwise it will return "0:0:0:0:0:0".

### Parameters

**list** Device list object

**index** Device index

**error** Pointer to an error object that will be set if an error occurs.

### Returns

const char\* returns the device mac address

Referenced by [ob::DeviceList::getLocalMacAddress\(\)](#).

## ◆ [ob\\_device\\_list\\_get\\_device\(\)](#)

```
OB_EXPORT ob_device * ob_device_list_get_device ( const ob_device_list* list,
                                                       uint32_t index,
                                                       ob_error ** error )
```

Create a device.

### Attention

If the device has already been acquired and created elsewhere, repeated acquisitions will return an error.

### Parameters

[in] **list** Device list object.

[in] **index** The index of the device to create.

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_device**\* The created device.

Referenced by [ob::DeviceList::getDevice\(\)](#).

◆ **ob\_device\_list\_get\_device\_by\_serial\_number()**

```
OB_EXPORT ob_device * ob_device_list_get_device_by_serial_number ( const ob_device_list * list,
                                                               const char *           serial_number,
                                                               ob_error **            error )
```

Create a device.

**Attention**

If the device has already been acquired and created elsewhere, repeated acquisitions will return an error.

**Parameters**

[in] **list** Device list object.

[in] **serial\_number** The serial number of the device to create.

[out] **error** Pointer to an error object that will be set if an error occurs.

**Returns**

**ob\_device\*** The created device.

Referenced by **ob::DeviceList::getDeviceBySN()**.

◆ **ob\_device\_list\_get\_device\_by\_uid()**

## Create device by uid.

On Linux platform, for usb device, the uid of the device is composed of bus-port-dev, for example 1-1.2-1. But the SDK will remove the dev number and only keep the bus-port as the uid to create the device, for example 1-1.2, so that we can create a device connected to the specified USB port. Similarly, users can also directly pass in bus-port as uid to create device.

For GMSL device, the uid is GMSL port with "gmsl2-" prefix, for example gmsl2-1.

## Attention

If the device has already been acquired and created elsewhere, repeated acquisitions will return an error.

## Parameters

[in] **list** Device list object.

[in] **uid** The UID of the device to create.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_device\*** The created device.

Referenced by [ob::DeviceList::getDeviceByUid\(\)](#).

- #### ◆ ob\_device\_get\_calibration\_camera\_param\_list()

Get the original parameter list of camera calibration saved on the device.

## Attention

The parameters in the list do not correspond to the current open-stream configuration. You need to select the parameters according to the actual situation, and may need to do scaling, mirroring and other processing. Non-professional users are recommended to use the [ob\\_pipeline\\_get\\_camera\\_param\(\)](#) interface.

## Parameters

[in] **device** The device object.

[out] **error** Log error messages.

## Returns

**ob camera param list** The camera parameter list.

Referenced by [ob::Device::getCalibrationCameraParamList\(\)](#).

- #### ◆ ob\_camera\_param\_list\_get\_count()

Get the number of camera parameter lists.

## Parameters

[in] **param list** Camera parameter list

[out] **error** Log error messages

## Returns

uint32\_t The number of lists

Referenced by [ob::CameraParamList::getCount\(\)](#).

- #### ◆ ob camera param list get param()

Get camera parameters from the camera parameter list.

## Parameters

[in] **param\_list** Camera parameter list

[in] **index** Parameter index

[out] **error** Log error messages

## Returns

**ob\_camera\_param** The camera parameters. Since it returns the structure object directly, there is no need to provide a delete interface.

Referenced by [ob::CameraParamList::getCameraParam\(\)](#).

#### ◆ ob\_delete\_camera\_param\_list()

Delete the camera parameter list.

## Parameters

[in] **param\_list** Camera parameter list

[out] **error** Log error messages

Referenced by [ob::CameraParamList::~CameraParamList\(\)](#).

# Device.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #ifndef __cplusplus
11 extern "C" {
12 #endif
13
14 #include "ObTypes.h"
15 #include "Property.h"
16 #include "MultipleDevices.h"
17 #include "Advanced.h"
18
25 OB_EXPORT void ob_delete_device(ob_device *device, ob_error **error);
26
34 OB_EXPORT ob_sensor_list *ob_device_get_sensor_list(const ob_device *device, ob_error **error);
35
44 OB_EXPORT ob_sensor *ob_device_get_sensor(ob_device *device, ob_sensor_type type, ob_error **er
ror);
45
54 OB_EXPORT void ob_device_set_int_property(ob_device *device, ob_property_id property_id, int32_t v
alue, ob_error **error);
55
64 OB_EXPORT int32_t ob_device_get_int_property(ob_device *device, ob_property_id property_id, ob_err
or **error);
65
74 OB_EXPORT ob_int_property_range ob_device_get_int_property_range(ob_device *device, ob_prop
erty_id property_id, ob_error **error);
75
84 OB_EXPORT void ob_device_set_float_property(ob_device *device, ob_property_id property_id, float v
alue, ob_error **error);
85
94 OB_EXPORT float ob_device_get_float_property(ob_device *device, ob_property_id property_id, ob_err
or **error);
95
104 OB_EXPORT ob_float_property_range ob_device_get_float_property_range(ob_device *device, ob_pro
perty_id property_id, ob_error **error);
105
114 OB_EXPORT void ob_device_set_bool_property(ob_device *device, ob_property_id property_id, bool v
alue, ob_error **error);
115
124 OB_EXPORT bool ob_device_get_bool_property(ob_device *device, ob_property_id property_id, ob_err
or **error);
125
134 OB_EXPORT ob_bool_property_range ob_device_get_bool_property_range(ob_device *device, ob_pro
perty_id property_id, ob_error **error);
135
145 OB_EXPORT void ob_device_set_structured_data(ob_device *device, ob_property_id property_id, cons
t uint8_t *data, uint32_t data_size, ob_error **error);
146
156 OB_EXPORT void ob_device_get_structured_data(ob_device *device, ob_property_id property_id, uint8
_t *data, uint32_t *data_size, ob_error **error);
157
167 OB_EXPORT void ob_device_get_raw_data(ob_device *device, ob_property_id property_id, ob_get_dat
a_callback cb, void *user_data, ob_error **error);
168
177 OB_EXPORT void ob_device_write_customer_data(ob_device *device, const void *data, uint32_t data_s
```

```

    -ize, ob_error **error);
178 OB_EXPORT void ob_device_read_customer_data(ob_device *device, void *data, uint32_t *data_size, ob_error **error);
188 OB_EXPORT uint32_t ob_device_get_supported_property_count(const ob_device *device, ob_error **error);
196 OB_EXPORT ob_property_item ob_device_get_supported_property_item(const ob_device *device, uint32_t index, ob_error **error);
206 OB_EXPORT bool ob_device_is_property_supported(const ob_device *device, ob_property_id property_id, ob_permission_type permission, ob_error **error);
217 OB_EXPORT bool ob_device_is_global_timestamp_supported(const ob_device *device, ob_error **error);
226 OB_EXPORT void ob_device_enable_global_timestamp(ob_device *device, bool enable, ob_error **error);
227 OB_EXPORT void ob_device_update_firmware(ob_device *device, const char *path, ob_device_fw_update_callback callback, bool async, void *user_data, ob_error **error);
247 OB_EXPORT void ob_device_update_firmware_from_data(ob_device *device, const uint8_t *data, uint32_t data_size, ob_device_fw_update_callback callback, bool async, void *user_data, ob_error **error);
261 OB_EXPORT void ob_device_update_optional_depth_presets(ob_device *device, const char file_path_list[][OB_PATH_MAX], uint8_t path_count, ob_device_fw_update_callback callback, void *user_data, ob_error **error);
274 OB_EXPORT void ob_device_set_state_changed_callback(ob_device *device, ob_device_state_callback callback, void *user_data, ob_error **error);
275 OB_EXPORT void ob_device_reboot(ob_device *device, ob_error **error);
285 OB_EXPORT ob_device_state ob_device_get_device_state(const ob_device *device, ob_error **error);
295 OB_EXPORT void ob_device_send_and_receive_data(ob_device *device, const uint8_t *send_data, uint32_t send_data_size, uint8_t *receive_data, uint32_t *receive_data_size, ob_error **error);
305 OB_EXPORT void ob_device_set_state_changed_callback(ob_device *device, ob_device_state_callback callback, void *user_data, ob_error **error);
317 OB_EXPORT void ob_device_enable_heartbeat(ob_device *device, bool enable, ob_error **error);
318 OB_EXPORT void ob_device_info *ob_device_get_device_info(const ob_device *device, ob_error **error);
331 OB_EXPORT const char *ob_device_info_get_name(const ob_device_info *info, ob_error **error);
341 OB_EXPORT void ob_delete_device_info(ob_device_info *info, ob_error **error);
349 OB_EXPORT int ob_device_info_get_pid(const ob_device_info *info, ob_error **error);
358 OB_EXPORT const char *ob_device_info_get_vid(const ob_device_info *info, ob_error **error);
367 OB_EXPORT const char *ob_device_info_get_uid(const ob_device_info *info, ob_error **error);
376 OB_EXPORT const char *ob_device_info_get_serial_number(const ob_device_info *info, ob_error **error);
385 OB_EXPORT const char *ob_device_info_get_connection_type(const ob_device_info *info, ob_error **error);
394 OB_EXPORT const char *ob_device_info_get_firmware_version(const ob_device_info *info, ob_error **error);
403 OB_EXPORT const char *ob_device_info_get_firmware_version(const ob_device_info *info, ob_error **error);
412 OB_EXPORT const char *ob_device_info_get_firmware_version(const ob_device_info *info, ob_error **error);

```

```

413     error);
423 OB_EXPORT const char *ob_device_info_get_ip_address(const ob_device_info *info, ob_error **error)
424     ;
432 OB_EXPORT const char *ob_device_info_get_hardware_version(const ob_device_info *info, ob_error *
433     *error);
442 OB_EXPORT bool ob_device_is_extension_info_exist(const ob_device *device, const char *info_key, ob
443     _error **error);
453 OB_EXPORT const char *ob_device_get_extension_info(const ob_device *device, const char *info_key,
454     ob_error **error);
462 OB_EXPORT const char *ob_device_info_get_supported_min_sdk_version(const ob_device_info *info,
463     ob_error **error);
471 OB_EXPORT const char *ob_device_info_get_asicName(const ob_device_info *info, ob_error **error);
472 OB_EXPORT ob_device_type ob_device_info_get_device_type(const ob_device_info *info, ob_error **
473     error);
481 OB_EXPORT void ob_delete_device_list(ob_device_list *list, ob_error **error);
489 OB_EXPORT uint32_t ob_device_list_get_count(const ob_device_list *list, ob_error **error);
497 OB_EXPORT const char *ob_device_list_get_device_name(const ob_device_list *list, uint32_t index, ob
498     _error **error);
507 OB_EXPORT int ob_device_list_get_device_pid(const ob_device_list *list, uint32_t index, ob_error **err
508     or);
517 OB_EXPORT int ob_device_list_get_device_vid(const ob_device_list *list, uint32_t index, ob_error **err
518     or);
528 OB_EXPORT const char *ob_device_list_get_device_uid(const ob_device_list *list, uint32_t index, ob_e
529     rror **error);
537 OB_EXPORT const char *ob_device_list_get_device_serial_number(const ob_device_list *list, uint32_t i
538     ndex, ob_error **error);
547 OB_EXPORT const char *ob_device_list_get_device_connection_type(const ob_device_list *list, uint32
548     _t index, ob_error **error);
558 OB_EXPORT const char *ob_device_list_get_device_ip_address(const ob_device_list *list, uint32_t ind
559     ex, ob_error **error);
570 OB_EXPORT const char *ob_device_list_get_device_local_mac(const ob_device_list *list, uint32_t inde
571     x, ob_error **error);
582 OB_EXPORT ob_device *ob_device_list_get_device(const ob_device_list *list, uint32_t index, ob_error
583     **error);
595 OB_EXPORT ob_device *ob_device_list_get_device_by_serial_number(const ob_device_list *list, const
596     char *serial_number, ob_error **error);
607 OB_EXPORT ob_device *ob_device_list_get_device_by_uid(const ob_device_list *list, const char *uid,
608     ob_error **error);
623 OB_EXPORT ob_camera_param_list *ob_device_get_calibration_camera_param_list(ob_device *device,
624     ob_error **error);
637 OB_EXPORT ob_camera_param_list *ob_camera_param_list_get_count(ob_camera_param_list *param_list, ob_error **e
638     rror);
646 OB_EXPORT uint32_t ob_camera_param_list_get_count(ob_camera_param_list *param_list, ob_error **e
647     rror);
656 OB_EXPORT ob_camera_param_list *ob_camera_param_list_get_param(ob_camera_param_list *param_list, ob_error **e
657     rror);

```

```

550 OB_EXPORT ob_camera_param ob_camera_param_list_get_param(ob_camera_param_list *param_list, ui
      nt32_t index, ob_error **error);
551
552
553 // The following interfaces are deprecated and are retained here for compatibility purposes.
554 #define ob_device_list_device_count ob_device_list_get_count
555 #define ob_device_list_get_extension_info ob_device_info_get_extension_info
556 #define ob_device_upgrade ob_device_update_firmware
557 #define ob_device_upgrade_from_data ob_device_update_firmware_from_data
558 #define ob_device_get_supported_property ob_device_get_supported_property_item
559 #define ob_device_state_changed ob_device_set_state_changed_callback
560 #define ob_device_info_name ob_device_info_get_name
561 #define ob_device_info_pid ob_device_info_get_pid
562 #define ob_device_info_vid ob_device_info_get_vid
563 #define ob_device_info_uid ob_device_info_get_uid
564 #define ob_device_info_serial_number ob_device_info_get_serial_number
565 #define ob_device_info_firmware_version ob_device_info_get_firmware_version
566 #define ob_device_info_connection_type ob_device_info_get_connection_type
567 #define ob_device_info_ip_address ob_device_info_get_ip_address
568 #define ob_device_info_hardware_version ob_device_info_get_hardware_version
569 #define ob_device_info_supported_min_sdk_version ob_device_info_get_supported_min_sdk_version
570 #define ob_device_info_asicName ob_device_info_get_asicName
571 #define ob_device_info_device_type ob_device_info_get_device_type
572 #define ob_device_list_get_device_count ob_device_list_get_count
573 #define ob_camera_param_list_count ob_camera_param_list_get_count
574
575 #ifdef __cplusplus
576 }
577 #endif
578
579
580
581
582
583
584
585
586
587
588
589
590
591

```

# Device.hpp File Reference

Device related types, including operations such as getting and creating a device, setting and obtaining device attributes, and obtaining sensors. [More...](#)

```
#include "Types.hpp"
#include "libobsensor/h/Property.h"
#include "libobsensor/h/Device.h"
#include "libobsensor/h/Advanced.h"
#include "libobsensor/hpp/Filter.hpp"
#include "libobsensor/hpp/Sensor.hpp"
#include "Error.hpp"
#include <memory>
#include <string>
#include <vector>
#include <thread>
#include <chrono>
```

[Go to the source code of this file.](#)

## Classes

class **ob::Device**

class **ob::DeviceInfo**

A class describing device information, representing the name, id, serial number and other basic information of an RGBD camera. [More...](#)

class **ob::DeviceList**

Class representing a list of devices. [More...](#)

class **ob::OBDepthWorkModeList**

class **ob::DevicePresetList**

Class representing a list of device presets. [More...](#)

class **ob::CameraParamList**

Class representing a list of camera parameters. [More...](#)

class **ob::DeviceFrameInterleaveList**

Class representing a list of device **Frame** Interleave. [More...](#)

class **ob::PresetResolutionConfigList**

Class representing a list of device **Frame** Interleave. [More...](#)

## Namespaces

namespace **ob**

## Macros

```
#define timerReset timestampReset
```

Alias for timestampReset since it is more accurate.

## Detailed Description

Device related types, including operations such as getting and creating a device, setting and obtaining device attributes, and obtaining sensors.

Definition in file [Device.hpp](#).

## Macro Definition Documentation

### ◆ timerReset

```
#define timerReset timestampReset
```

Alias for timestampReset since it is more accurate.

Definition at line [643](#) of file [Device.hpp](#).

# Device.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5 #include "Types.hpp"
6
7 #include "libobsensor/h/Property.h"
8 #include "libobsensor/h/Device.h"
9 #include "libobsensor/h/Advanced.h"
10 #include "libobsensor/hpp/Filter.hpp"
11 #include "libobsensor/hpp/Sensor.hpp"
12
13 #include "Error.hpp"
14 #include <memory>
15 #include <string>
16 #include <vector>
17 #include <thread>
18 #include <chrono>
19
20 namespace ob {
21
22 class DeviceInfo;
23 class SensorList;
24 class DevicePresetList;
25 class OBDepthWorkModeList;
26 class CameraParamList;
27 class DeviceFrameInterleaveList;
28 class PresetResolutionConfigList;
29
30 class Device {
31 public:
32     typedef std::function<void(OBFwUpdateState state, const char *message, uint8_t percent)> DeviceFwUpdateCallback;
33     typedef std::function<void(OBDeviceState state, const char *message)> DeviceStateChangedCallback;
34 protected:
35     ob_device_t *impl_ = nullptr;
36     DeviceStateChangedCallback deviceStateChangeCallback_;
37     DeviceFwUpdateCallback fwUpdateCallback_;
38
39 public:
40     explicit Device(ob_device_t *impl) : impl_(impl) {}
41
42     Device(Device &&other) noexcept : impl_(other.impl_) {
43         other.impl_ = nullptr;
44     }
45
46     Device &operator=(Device &&other) noexcept {
47         if(this != &other) {
48             ob_error *error = nullptr;
49             ob_delete_device(impl_, &error);
50             Error::handle(&error);
51             impl_ = other.impl_;
52             other.impl_ = nullptr;
53         }
54         return *this;
55     }
56 }
```

```

79
80     Device(const Device &) = delete;
81     Device &operator=(const Device &) = delete;
82
83     virtual ~Device() noexcept {
84         ob_error *error = nullptr;
85         ob_delete_device(impl_, &error);
86         Error::handle(&error, false);
87     }
88
89     ob_device_t *getImpl() const {
90         return impl_;
91     }
92
93     std::shared_ptr<DeviceInfo> getDeviceInfo() const {
94         ob_error *error = nullptr;
95         auto info = ob_device_get_device_info(impl_, &error);
96         Error::handle(&error);
97         return std::make_shared<DeviceInfo>(info);
98     }
99
100    bool isExtensionInfoExist(const std::string &infoKey) const {
101        ob_error *error = nullptr;
102        auto exist = ob_device_is_extension_info_exist(impl_, infoKey.c_str(), &error);
103        Error::handle(&error);
104        return exist;
105    }
106
107
108    const char *getExtensionInfo(const std::string &infoKey) const {
109        ob_error *error = nullptr;
110        const char *info = ob_device_get_extension_info(impl_, infoKey.c_str(), &error);
111        Error::handle(&error);
112        return info;
113    }
114
115
116    std::shared_ptr<SensorList> getSensorList() const {
117        ob_error *error = nullptr;
118        auto list = ob_device_get_sensor_list(impl_, &error);
119        Error::handle(&error);
120        return std::make_shared<SensorList>(list);
121    }
122
123
124    std::shared_ptr<Sensor> getSensor(OBSensorType type) const {
125        ob_error *error = nullptr;
126        auto sensor = ob_device_get_sensor(impl_, type, &error);
127        Error::handle(&error);
128        return std::make_shared<Sensor>(sensor);
129    }
130
131
132    void setIntProperty(OBPropertyID propertyId, int32_t value) const {
133        ob_error *error = nullptr;
134        ob_device_set_int_property(impl_, propertyId, value, &error);
135        Error::handle(&error);
136    }
137
138
139    void setFloatProperty(OBPropertyID propertyId, float value) const {
140        ob_error *error = nullptr;
141        ob_device_set_float_property(impl_, propertyId, value, &error);
142        Error::handle(&error);
143    }
144
145
146    void setBoolProperty(OBPropertyID propertyId, bool value) const {
147        ob_error *error = nullptr;
148        ob_device_set_bool_property(impl_, propertyId, value, &error);
149        Error::handle(&error);
150    }

```

```

190    }
191
198 int32_t getIntProperty(OBPropertyID propertyId) const {
199     ob_error *error = nullptr;
200     auto value = ob_device_get_int_property(impl_, propertyId, &error);
201     Error::handle(&error);
202     return value;
203 }
204
211 float getFloatProperty(OBPropertyID propertyId) const {
212     ob_error *error = nullptr;
213     auto value = ob_device_get_float_property(impl_, propertyId, &error);
214     Error::handle(&error);
215     return value;
216 }
217
224 bool getBoolProperty(OBPropertyID propertyId) const {
225     ob_error *error = nullptr;
226     auto value = ob_device_get_bool_property(impl_, propertyId, &error);
227     Error::handle(&error);
228     return value;
229 }
230
237 OBIntPropertyRange getIntPropertyRange(OBPropertyID propertyId) const {
238     ob_error *error = nullptr;
239     auto range = ob_device_get_int_property_range(impl_, propertyId, &error);
240     Error::handle(&error);
241     return range;
242 }
243
250 OBFloatPropertyRange getFloatPropertyRange(OBPropertyID propertyId) const {
251     ob_error *error = nullptr;
252     auto range = ob_device_get_float_property_range(impl_, propertyId, &error);
253     Error::handle(&error);
254     return range;
255 }
256
263 OBBBoolPropertyRange getBoolPropertyRange(OBPropertyID propertyId) const {
264     ob_error *error = nullptr;
265     auto range = ob_device_get_bool_property_range(impl_, propertyId, &error);
266     Error::handle(&error);
267     return range;
268 }
269
277 void setStructuredData(OBPropertyID propertyId, const uint8_t *data, uint32_t dataSize) const {
278     ob_error *error = nullptr;
279     ob_device_set_structured_data(impl_, propertyId, data, dataSize, &error);
280     Error::handle(&error);
281 }
282
290 void getStructuredData(OBPropertyID propertyId, uint8_t *data, uint32_t *dataSize) const {
291     ob_error *error = nullptr;
292     ob_device_get_structured_data(impl_, propertyId, data, dataSize, &error);
293     Error::handle(&error);
294 }
295
302 void writeCustomerData(const void *data, uint32_t dataSize) {
303     ob_error *error = nullptr;
304     ob_device_write_customer_data(impl_, data, dataSize, &error);
305     Error::handle(&error);
306 }
307
314 void readCustomerData(void *data, uint32_t *dataSize) {
315     ob_error *error = nullptr;
316     ob_device_read_customer_data(impl_, data, dataSize, &error);
317 }
```

```

31 /     Error::handle(&error);
318 } 
319
325 int getSupportedPropertyCount() const {
326     ob_error *error = nullptr;
327     auto count = ob_device_get_supported_property_count(impl_, &error);
328     Error::handle(&error);
329     return count;
330 }
331
338 OBPropertyItem getSupportedProperty(uint32_t index) const {
339     ob_error *error = nullptr;
340     auto item = ob_device_get_supported_property_item(impl_, index, &error);
341     Error::handle(&error);
342     return item;
343 }
344
352 bool isPropertySupported(OBPropertyID propertyId, OBPermissionType permission) const {
353     ob_error *error = nullptr;
354     auto result = ob_device_is_property_supported(impl_, propertyId, permission, &error);
355     Error::handle(&error);
356     return result;
357 }
358
364 bool isGlobalTimestampSupported() const {
365     ob_error *error = nullptr;
366     auto result = ob_device_is_global_timestamp_supported(impl_, &error);
367     Error::handle(&error);
368     return result;
369 }
370
376 void enableGlobalTimestamp(bool enable) {
377     ob_error *error = nullptr;
378     ob_device_enable_global_timestamp(impl_, enable, &error);
379     Error::handle(&error);
380 }
381
389 void updateFirmware(const char *filePath, DeviceFwUpdateCallback callback, bool async = true) {
390     ob_error *error = nullptr;
391     fwUpdateCallback_ = callback;
392     ob_device_update_firmware(impl_, filePath, &Device::firmwareUpdateCallback, async, this, &error);
393     Error::handle(&error);
394 }
395
404 void updateFirmwareFromData(const uint8_t *firmwareData, uint32_t firmwareDataSize, DeviceFwUpdat
    eCallback callback, bool async = true) {
405     ob_error *error = nullptr;
406     fwUpdateCallback_ = callback;
407     ob_device_update_firmware_from_data(impl_, firmwareData, firmwareDataSize, &Device::firmwareUp
    dateCallback, async, this, &error);
408     Error::handle(&error);
409 }
410
418 void updateOptionalDepthPresets(const char filePathList[][OB_PATH_MAX], uint8_t pathCount, Devi
    ceFwUpdateCallback callback) {
419     ob_error *error = nullptr;
420     fwUpdateCallback_ = callback;
421     ob_device_update_optional_depth_presets(impl_, filePathList, pathCount, &Device::firmwareUpdate
    Callback, this, &error);
422     Error::handle(&error);
423 }
424
431 void setDeviceStateChangedCallback(DeviceStateChangedCallback callback) {
432     ob_error *error = nullptr;
433     deviceStateChangeCallback_ = callback;
434     ob_device_set_state_changed_callback(impl_, &Device::deviceStateChangedCallback, this, &error);

```

```

435     Error::handle(&error);
436 }
437
438 static void deviceStateChangedCallback(OBDeviceState state, const char *message, void *userData) {
439     auto device = static_cast<Device *>(userData);
440     device->deviceStateChangeCallback_(state, message);
441 }
442
443 OBDepthWorkMode getCurrentDepthWorkMode() const {
444     ob_error *error = nullptr;
445     auto mode = ob_device_get_current_depth_work_mode(impl_, &error);
446     Error::handle(&error);
447     return mode;
448 }
449
450 const char *getCurrentDepthModeName() {
451     ob_error *error = nullptr;
452     auto name = ob_device_get_current_depth_work_mode_name(impl_, &error);
453     Error::handle(&error);
454     return name;
455 }
456
457 OBStatus switchDepthWorkMode(const OBDepthWorkMode &workMode) const {
458     ob_error *error = nullptr;
459     auto status = ob_device_switch_depth_work_mode(impl_, &workMode, &error);
460     Error::handle(&error);
461     return status;
462 }
463
464 OBStatus switchDepthWorkMode(const char *modeName) const {
465     ob_error *error = nullptr;
466     auto status = ob_device_switch_depth_work_mode_by_name(impl_, modeName, &error);
467     Error::handle(&error);
468     return status;
469 }
470
471 std::shared_ptr<OBDepthWorkModeList> getDepthWorkModeList() const {
472     ob_error *error = nullptr;
473     auto list = ob_device_get_depth_work_mode_list(impl_, &error);
474     Error::handle(&error);
475     return std::make_shared<OBDepthWorkModeList>(list);
476 }
477
478 void reboot() const {
479     ob_error *error = nullptr;
480     ob_device_reboot(impl_, &error);
481     Error::handle(&error);
482 }
483
484 void reboot(uint32_t delayMs) const {
485     setIntProperty(OB_PROP_DEVICE_REBOOT_DELAY_INT, delayMs);
486     reboot();
487 }
488
489 void enableHeartbeat(bool enable) const {
490     ob_error *error = nullptr;
491     ob_device_enable_heartbeat(impl_, enable, &error);
492     Error::handle(&error);
493 }
494
495 uint16_t getSupportedMultiDeviceSyncModeBitmap() const {
496     ob_error *error = nullptr;
497     auto mode = ob_device_get_supported_multi_device_sync_mode_bitmap(impl_, &error);
498     Error::handle(&error);
499     return mode;
500 }
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559

```

```

560     }
561
567 void setMultiDeviceSyncConfig(const OBMMultiDeviceSyncConfig &config) const {
568     ob_error *error = nullptr;
569     ob_device_set_multi_device_sync_config(impl_, &config, &error);
570     Error::handle(&error);
571 }
572
578 OBMMultiDeviceSyncConfig getMultiDeviceSyncConfig() const {
579     ob_error *error = nullptr;
580     auto config = ob_device_get_multi_device_sync_config(impl_, &error);
581     Error::handle(&error);
582     return config;
583 }
584
596 void triggerCapture() const {
597     ob_error *error = nullptr;
598     ob_device_trigger_capture(impl_, &error);
599     Error::handle(&error);
600 }
601
605 void setTimestampResetConfig(const OBDDeviceTimestampResetConfig &config) const {
606     ob_error *error = nullptr;
607     ob_device_set_timestamp_reset_config(impl_, &config, &error);
608     Error::handle(&error);
609 }
610
616 OBDDeviceTimestampResetConfig getTimestampResetConfig() const {
617     ob_error *error = nullptr;
618     auto config = ob_device_get_timestamp_reset_config(impl_, &error);
619     Error::handle(&error);
620     return config;
621 }
622
634 void timestampReset() const {
635     ob_error *error = nullptr;
636     ob_device_timestamp_reset(impl_, &error);
637     Error::handle(&error);
638 }
639
643 #define timerReset timestampReset
644
656 void timerSyncWithHost() const {
657     ob_error *error = nullptr;
658     ob_device_timer_sync_with_host(impl_, &error);
659     Error::handle(&error);
660 }
661
667 const char *getCurrentPresetName() const {
668     ob_error *error = nullptr;
669     const char *name = ob_device_get_current_preset_name(impl_, &error);
670     Error::handle(&error);
671     return name;
672 }
673
680 void loadPreset(const char *presetName) const {
681     ob_error *error = nullptr;
682     ob_device_load_preset(impl_, presetName, &error);
683     Error::handle(&error);
684 }
685
693 std::shared_ptr<DevicePresetList> getAvailablePresetList() const {
694     ob_error *error = nullptr;
695     auto list = ob_device_get_available_preset_list(impl_, &error);
696     Error::handle(&error);
697 }

```

```

697     return std::make_shared<DevicePresetList>(list);
698 }
699
700 void loadPresetFromJsonFile(const char *filePath) const {
701     ob_error *error = nullptr;
702     ob_device_load_preset_from_json_file(impl_, filePath, &error);
703     Error::handle(&error);
704 }
705
727 void loadPresetFromJsonData(const char *presetName, const uint8_t *data, uint32_t size) {
728     ob_error *error = nullptr;
729     ob_device_load_preset_from_json_data(impl_, presetName, data, size, &error);
730 }
731
743 void exportSettingsAsPresetJsonData(const char *presetName, const uint8_t **data, uint32_t *dataSize
744 e) {
745     ob_error *error = nullptr;
746     ob_device_export_current_settings_as_preset_json_data(impl_, presetName, data, dataSize, &error);
747 }
755 void exportSettingsAsPresetJsonFile(const char *filePath) const {
756     ob_error *error = nullptr;
757     ob_device_export_current_settings_as_preset_json_file(impl_, filePath, &error);
758     Error::handle(&error);
759 }
760
766 OBDeviceState getDeviceState() {
767     OBDeviceState state = {};
768     ob_error *error = nullptr;
769     state = ob_device_get_device_state(impl_, &error);
770     return state;
771 }
772
785 void sendAndReceiveData(const uint8_t *sendData, uint32_t sendDataSize, uint8_t *receiveData, uint3
786 2_t *receiveDataSize) const {
787     ob_error *error = nullptr;
788     ob_device_send_and_receive_data(impl_, sendData, sendDataSize, receiveData, receiveDataSize, &er
789 rror);
790     Error::handle(&error);
791 }
792
796 bool isFrameInterleaveSupported() const {
797     ob_error *error = nullptr;
798     bool ret = ob_device_is_frame_interleave_supported(impl_, &error);
799     Error::handle(&error);
800     return ret;
801 }
802
808 void loadFrameInterleave(const char *frameInterleaveName) const {
809     ob_error *error = nullptr;
810     ob_device_load_frame_interleave(impl_, frameInterleaveName, &error);
811     Error::handle(&error);
812 }
813
819 std::shared_ptr<DeviceFrameInterleaveList> getAvailableFrameInterleaveList() const {
820     ob_error *error = nullptr;
821     auto list = ob_device_get_available_frame_interleave_list(impl_, &error);
822     Error::handle(&error);
823     return std::make_shared<DeviceFrameInterleaveList>(list);
824 }
825
831 std::shared_ptr<PresetResolutionConfigList> getAvailablePresetResolutionConfigList() const {
832     ob_error *error = nullptr;
833     auto list = ob_device_get_available_preset_resolution_config_list(impl_, &error);
834     Error::handle(&error);
835     return std::make_shared<PresetResolutionConfigList>(list);

```

```

835     return std::make_shared<ResourceResolutionConfigList>(list),
836 }
837
838 private:
839     static void firmwareUpdateCallback(ob_fw_update_state state, const char *message, uint8_t percent, v
840         oid *userData) {
841         auto device = static_cast<Device *>(userData);
842         if(device && device->fwUpdateCallback_) {
843             device->fwUpdateCallback_(state, message, percent);
844         }
845     }
846
847 public:
848     // The following interfaces are deprecated and are retained here for compatibility purposes.
849     void deviceUpgrade(const char *filePath, DeviceFwUpdateCallback callback, bool async = true) {
850         updateFirmware(filePath, callback, async);
851     }
852     void deviceUpgradeFromData(const uint8_t *firmwareData, uint32_t firmwareDataSize, DeviceFwUpdat
853         eCallback callback, bool async = true) {
854         updateFirmwareFromData(firmwareData, firmwareDataSize, callback, async);
855     }
856     std::shared_ptr<CameraParamList> getCalibrationCameraParamList() {
857         ob_error *error = nullptr;
858         auto impl = ob_device_get_calibration_camera_param_list(impl_, &error);
859         Error::handle(&error);
860         return std::make_shared<CameraParamList>(impl);
861     }
862     void loadDepthFilterConfig(const char *filePath) {
863         // In order to compile, some high-version compilers will warn that the function parameters are not used
864
865         (void)filePath;
866     };
867 };
868
872 class DeviceInfo {
873 private:
874     ob_device_info_t *impl_ = nullptr;
875
876 public:
877     explicit DeviceInfo(ob_device_info_t *impl) : impl_(impl) {}
878     ~DeviceInfo() noexcept {
879         ob_error *error = nullptr;
880         ob_delete_device_info(impl_, &error);
881         Error::handle(&error, false);
882     }
883
889     const char *getName() const {
890         ob_error *error = nullptr;
891         const char *name = ob_device_info_get_name(impl_, &error);
892         Error::handle(&error);
893         return name;
894     }
895
901     int getPid() const {
902         ob_error *error = nullptr;
903         int pid = ob_device_info_get_pid(impl_, &error);
904         Error::handle(&error);
905         return pid;
906     }
907
913     int getVid() const {
914         ob_error *error = nullptr;
915         int vid = ob_device_info_get_vid(impl_, &error);

```

```
916     Error::handle(&error);
917     return vid;
918 }
919
925 const char *getUid() const {
926     ob_error *error=nullptr;
927     const char *uid = ob_device_info_get_uid(impl_, &error);
928     Error::handle(&error);
929     return uid;
930 }
931
937 const char *getSerialNumber() const {
938     ob_error *error=nullptr;
939     const char *sn = ob_device_info_get_serial_number(impl_, &error);
940     Error::handle(&error);
941     return sn;
942 }
943
949 const char *getFirmwareVersion() const {
950     ob_error *error = nullptr;
951     const char *version = ob_device_info_get_firmware_version(impl_, &error);
952     Error::handle(&error);
953     return version;
954 }
955
962 const char *getConnectionType() const {
963     ob_error *error = nullptr;
964     const char *type = ob_device_info_get_connection_type(impl_, &error);
965     Error::handle(&error);
966     return type;
967 }
968
976 const char *getIpAddress() const {
977     ob_error *error = nullptr;
978     const char *ip = ob_device_info_get_ip_address(impl_, &error);
979     Error::handle(&error);
980     return ip;
981 }
982
988 const char *getHardwareVersion() const {
989     ob_error *error = nullptr;
990     const char *version = ob_device_info_get_hardware_version(impl_, &error);
991     Error::handle(&error);
992     return version;
993 }
994
1000 const char *getSupportedMinSdkVersion() const {
1001     ob_error *error = nullptr;
1002     const char *version = ob_device_info_get_supported_min_sdk_version(impl_, &error);
1003     Error::handle(&error);
1004     return version;
1005 }
1006
1012 const char *getAsicName() const {
1013     ob_error *error = nullptr;
1014     const char *name = ob_device_info_get_asicName(impl_, &error);
1015     Error::handle(&error);
1016     return name;
1017 }
1018
1024 OBDeviceType getDeviceType() const {
1025     ob_error *error = nullptr;
1026     OBDeviceType type = ob_device_info_get_device_type(impl_, &error);
1027     Error::handle(&error);
1028     return type;
```

```
1029     }
1030
1031 public:
1032     // The following interfaces are deprecated and are retained here for compatibility purposes.
1033     const char *name() const {
1034         return getName();
1035     }
1036
1037     int pid() const {
1038         return getPid();
1039     }
1040
1041     int vid() const {
1042         return getVid();
1043     }
1044
1045     const char *uid() const {
1046         return getUid();
1047     }
1048
1049     const char *serialNumber() const {
1050         return getSerialNumber();
1051     }
1052
1053     const char *firmwareVersion() const {
1054         return getFirmwareVersion();
1055     }
1056
1057     const char *connectionType() const {
1058         return getConnectionType();
1059     }
1060
1061     const char *ipAddress() const {
1062         return getIpAddress();
1063     }
1064
1065     const char *hardwareVersion() const {
1066         return getHardwareVersion();
1067     }
1068
1069     const char *supportedMinSdkVersion() const {
1070         return getSupportedMinSdkVersion();
1071     }
1072
1073     const char *asicName() const {
1074         return getAsicName();
1075     }
1076
1077     OBDeviceType deviceType() const {
1078         return getDeviceType();
1079     }
1080 };
1081
1082 class DeviceList {
1083 private:
1084     ob_device_list_t *impl_ = nullptr;
1085
1086 public:
1087     explicit DeviceList(ob_device_list_t *impl) : impl_(impl) {}
1088     ~DeviceList() noexcept {
1089         ob_error *error = nullptr;
1090         ob_delete_device_list(impl_, &error);
1091         Error::handle(&error, false);
1092     }
1093 }
```

```

1102     uint32_t getCount() const {
1103         ob_error *error = nullptr;
1104         auto count = ob_device_list_get_count(impl_, &error);
1105         Error::handle(&error);
1106         return count;
1107     }
1108
1115     int getPid(uint32_t index) const {
1116         ob_error *error = nullptr;
1117         auto pid = ob_device_list_get_device_pid(impl_, index, &error);
1118         Error::handle(&error);
1119         return pid;
1120     }
1121
1128     int getVid(uint32_t index) const {
1129         ob_error *error = nullptr;
1130         auto vid = ob_device_list_get_device_vid(impl_, index, &error);
1131         Error::handle(&error);
1132         return vid;
1133     }
1134
1141     const char *getUid(uint32_t index) const {
1142         ob_error *error = nullptr;
1143         auto uid = ob_device_list_get_device_uid(impl_, index, &error);
1144         Error::handle(&error);
1145         return uid;
1146     }
1147
1154     const char *getSerialNumber(uint32_t index) const {
1155         ob_error *error = nullptr;
1156         auto sn = ob_device_list_get_device_serial_number(impl_, index, &error);
1157         Error::handle(&error);
1158         return sn;
1159     }
1160
1170     const char *getName(uint32_t index) const {
1171         ob_error *error = nullptr;
1172         auto name = ob_device_list_get_device_name(impl_, index, &error);
1173         Error::handle(&error);
1174         return name;
1175     }
1176
1183     const char *getConnectionType(uint32_t index) const {
1184         ob_error *error = nullptr;
1185         auto type = ob_device_list_get_device_connection_type(impl_, index, &error);
1186         Error::handle(&error);
1187         return type;
1188     }
1189
1198     const char *getIpAddress(uint32_t index) const {
1199         ob_error *error = nullptr;
1200         auto ip = ob_device_list_get_device_ip_address(impl_, index, &error);
1201         Error::handle(&error);
1202         return ip;
1203     }
1204
1213     const char *getLocalMacAddress(uint32_t index) const {
1214         ob_error *error = nullptr;
1215         auto mac = ob_device_list_get_device_local_mac(impl_, index, &error);
1216         Error::handle(&error);
1217         return mac;
1218     }
1219
1228     std::shared_ptr<Device> getDevice(uint32_t index) const {
1229         ob_error *error = nullptr;
1230         auto device = ob_device_list_get_device(impl_, index, &error);

```

```

1200     auto device = ob_device_list_get_device(impl_, index, &error);
1201     Error::handle(&error);
1202     return std::make_shared<Device>(device);
1203 }
1204
1205 std::shared_ptr<Device> getDeviceBySN(const char *serialNumber) const {
1206     ob_error *error = nullptr;
1207     auto device = ob_device_list_get_device_by_serial_number(impl_, serialNumber, &error);
1208     Error::handle(&error);
1209     return std::make_shared<Device>(device);
1210 }
1211
1212 std::shared_ptr<Device> getDeviceByUid(const char *uid) const {
1213     ob_error *error = nullptr;
1214     auto device = ob_device_list_get_device_by_uid(impl_, uid, &error);
1215     Error::handle(&error);
1216     return std::make_shared<Device>(device);
1217 }
1218
1219 public:
1220     // The following interfaces are deprecated and are retained here for compatibility purposes.
1221     uint32_t deviceCount() const {
1222         return getCount();
1223     }
1224
1225     int pid(uint32_t index) const {
1226         return getPid(index);
1227     }
1228
1229     int vid(uint32_t index) const {
1230         return getVid(index);
1231     }
1232
1233     const char *uid(uint32_t index) const {
1234         return getUid(index);
1235     }
1236
1237     const char *serialNumber(uint32_t index) const {
1238         return getSerialNumber(index);
1239     }
1240
1241     const char *name(uint32_t index) const {
1242         return getName(index);
1243     }
1244
1245     const char *connectionType(uint32_t index) const {
1246         return getConnectionType(index);
1247     }
1248
1249     const char *ipAddress(uint32_t index) const {
1250         return getIpAddress(index);
1251     }
1252 };
1253
1254 class OBDepthWorkModeList {
1255 private:
1256     ob_depth_work_mode_list_t *impl_ = nullptr;
1257
1258 public:
1259     explicit OBDepthWorkModeList(ob_depth_work_mode_list_t *impl) : impl_(impl) {}
1260     ~OBDepthWorkModeList() {
1261         ob_error *error = nullptr;
1262         ob_delete_depth_work_mode_list(impl_, &error);
1263         Error::handle(&error, false);
1264     }
1265

```

```

1321     uint32_t getCount() {
1322         ob_error *error = nullptr;
1323         auto count = ob_depth_work_mode_list_get_count(impl_, &error);
1324         Error::handle(&error);
1325         return count;
1326     }
1327
1328     OBDepthWorkMode getOBDepthWorkMode(uint32_t index) {
1329         ob_error *error = nullptr;
1330         auto mode = ob_depth_work_mode_list_get_item(impl_, index, &error);
1331         Error::handle(&error);
1332         return mode;
1333     }
1334
1335     OBDepthWorkMode operator[](uint32_t index) {
1336         return getOBDepthWorkMode(index);
1337     }
1338
1339 public:
1340     // The following interfaces are deprecated and are retained here for compatibility purposes.
1341     uint32_t count() {
1342         return getCount();
1343     }
1344
1345     class DevicePresetList {
1346 private:
1347     ob_device_preset_list_t *impl_ = nullptr;
1348
1349 public:
1350     explicit DevicePresetList(ob_device_preset_list_t *impl) : impl_(impl) {}
1351     ~DevicePresetList() noexcept {
1352         ob_error *error = nullptr;
1353         ob_delete_preset_list(impl_, &error);
1354         Error::handle(&error, false);
1355     }
1356
1357     uint32_t getCount() {
1358         ob_error *error = nullptr;
1359         auto count = ob_device_preset_list_get_count(impl_, &error);
1360         Error::handle(&error);
1361         return count;
1362     }
1363
1364     const char *getName(uint32_t index) {
1365         ob_error *error = nullptr;
1366         const char *name = ob_device_preset_list_get_name(impl_, index, &error);
1367         Error::handle(&error);
1368         return name;
1369     }
1370
1371     bool hasPreset(const char *name) {
1372         ob_error *error = nullptr;
1373         auto result = ob_device_preset_list_has_preset(impl_, name, &error);
1374         Error::handle(&error);
1375         return result;
1376     }
1377
1378 public:
1379     // The following interfaces are deprecated and are retained here for compatibility purposes.
1380     uint32_t count() {
1381         return getCount();
1382     }
1383
1384     };
1385
1386     };
1387
1388     };
1389
1390     };
1391
1392     };
1393
1394     };
1395
1396     };
1397
1398     };
1399
1400     };
1401
1402     };
1403
1404     };
1405
1406     };
1407
1408     };
1409
1410     };
1411
1412     };
1413
1414     };
1415
1416     };
1417

```

```

1421 class CameraParamList {
1422 private:
1423     ob_camera_param_list_t *impl_ = nullptr;
1424
1425 public:
1426     explicit CameraParamList(ob_camera_param_list_t *impl) : impl_(impl) {}
1427     ~CameraParamList() noexcept {
1428         ob_error *error = nullptr;
1429         ob_delete_camera_param_list(impl_, &error);
1430         Error::handle(&error, false);
1431     }
1432
1433     uint32_t getCount() {
1434         ob_error *error = nullptr;
1435         auto count = ob_camera_param_list_get_count(impl_, &error);
1436         Error::handle(&error);
1437         return count;
1438     }
1439
1440     OBCameraParam getCameraParam(uint32_t index) {
1441         ob_error *error = nullptr;
1442         auto param = ob_camera_param_list_get_param(impl_, index, &error);
1443         Error::handle(&error);
1444         return param;
1445     }
1446
1447 public:
1448     // The following interfaces are deprecated and are retained here for compatibility purposes.
1449     uint32_t count() {
1450         return getCount();
1451     }
1452 };
1453
1454 class DeviceFrameInterleaveList {
1455 private:
1456     ob_device_frame_interleave_list_t *impl_ = nullptr;
1457
1458 public:
1459     explicit DeviceFrameInterleaveList(ob_device_frame_interleave_list_t *impl) : impl_(impl) {}
1460     ~DeviceFrameInterleaveList() noexcept {
1461         ob_error *error = nullptr;
1462         ob_delete_frame_interleave_list(impl_, &error);
1463         Error::handle(&error, false);
1464     }
1465
1466     uint32_t getCount() {
1467         ob_error *error = nullptr;
1468         auto count = ob_device_frame_interleave_list_get_count(impl_, &error);
1469         Error::handle(&error);
1470         return count;
1471     }
1472
1473     const char *getName(uint32_t index) {
1474         ob_error *error = nullptr;
1475         const char *name = ob_device_frame_interleave_list_get_name(impl_, index, &error);
1476         Error::handle(&error);
1477         return name;
1478     }
1479
1480     bool hasFrameInterleave(const char *name) {
1481         ob_error *error = nullptr;
1482         auto result = ob_device_frame_interleave_list_has_frame_interleave(impl_, name, &error);
1483         Error::handle(&error);
1484         return result;
1485     }
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516

```

```

1510 };
1511
1512 class PresetResolutionConfigList {
1513 private:
1514     ob_preset_resolution_config_list_t *impl_ = nullptr;
1515
1516 public:
1517     explicit PresetResolutionConfigList(ob_preset_resolution_config_list_t *impl) : impl_(impl) {}
1518     ~PresetResolutionConfigList() noexcept {
1519         ob_error *error = nullptr;
1520         ob_delete_preset_resolution_config_list(impl_, &error);
1521         Error::handle(&error, false);
1522     }
1523
1524     uint32_t getCount() {
1525         ob_error *error = nullptr;
1526         auto count = ob_device_preset_resolution_config_get_count(impl_, &error);
1527         Error::handle(&error);
1528         return count;
1529     }
1530
1531     /*
1532      * @brief Get the device preset resolution ratio at the specified index
1533      * @param index the index of the device preset resolution ratio
1534      * @return OBPresetResolutionConfig the corresponding device preset resolution ratio
1535      */
1536     OBPresetResolutionConfig getPresetResolutionRatioConfig(uint32_t index) {
1537         ob_error *error = nullptr;
1538         auto config = ob_device_preset_resolution_config_list_get_item(impl_, index, &error);
1539         Error::handle(&error);
1540         return config;
1541     }
1542 };
1543
1544 } // namespace ob

```

# Error.h File Reference

Functions for handling errors, mainly used for obtaining error messages. [More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_error_status ob_error_get_status  
#define ob_error_message ob_error_get_message  
#define ob_error_function ob_error_get_function  
#define ob_error_args ob_error_get_args  
#define ob_error_exception_type ob_error_get_exception_type
```

## Functions

**OB\_EXPORT** **ob\_error** \* **ob\_create\_error** (**ob\_status** status, const char \*message, const char \*function, const char \*args, **ob\_exception\_type** exception\_type)  
Create a new error object.

**OB\_EXPORT** **ob\_status** **ob\_error\_get\_status** (const **ob\_error** \*error)  
Get the error status.

**OB\_EXPORT** const char \* **ob\_error\_get\_message** (const **ob\_error** \*error)  
Get the error message.

**OB\_EXPORT** const char \* **ob\_error\_get\_function** (const **ob\_error** \*error)  
Get the name of the API function that caused the error.

**OB\_EXPORT** const char \* **ob\_error\_get\_args** (const **ob\_error** \*error)  
Get the error parameters.

**OB\_EXPORT** **ob\_exception\_type** **ob\_error\_get\_exception\_type** (const **ob\_error** \*error)  
Get the type of exception that caused the error.

**OB\_EXPORT** void **ob\_delete\_error** (**ob\_error** \*error)  
Delete the error object.

## Detailed Description

Functions for handling errors, mainly used for obtaining error messages.

Definition in file **Error.h**.

## Macro Definition Documentation

- ◆ **ob\_error\_status**

```
#define ob_error_status ob_error_get_status
```

Definition at line **76** of file **Error.h**.

- ◆ **ob\_error\_message**

```
#define ob_error_message ob_error_get_message
```

Definition at line **77** of file **Error.h**.

- ◆ **ob\_error\_function**

```
#define ob_error_function ob_error_get_function
```

Definition at line **78** of file **Error.h**.

- ◆ **ob\_error\_args**

```
#define ob_error_args ob_error_get_args
```

Definition at line **79** of file **Error.h**.

- ◆ **ob\_error\_exception\_type**

```
#define ob_error_exception_type ob_error_get_exception_type
```

Definition at line **80** of file **Error.h**.

## Function Documentation

- ◆ **ob\_create\_error()**

```
OB_EXPORT ob_error* ob_create_error( ob_status status,  
                                     const char* message,  
                                     const char* function,  
                                     const char* args,  
                                     ob_exception_type exception_type )
```

Create a new error object.

#### Parameters

**status** The error status.  
**message** The error message.  
**function** The name of the API function that caused the error.  
**args** The error parameters.  
**exception\_type** The type of exception that caused the error.

#### Returns

ob\_error\* The new error object.

Referenced by [ob::StreamProfileFactory::create\(\)](#).

### ◆ [ob\\_error\\_get\\_status\(\)](#)

```
OB_EXPORT ob_status ob_error_get_status( const ob_error* error )
```

Get the error status.

#### Parameters

[in] **error** The error object.

#### Returns

The error status.

### ◆ [ob\\_error\\_get\\_message\(\)](#)

**OB\_EXPORT** const char \* ob\_error\_get\_message ( const **ob\_error** \* error )

Get the error message.

#### Parameters

[in] **error** The error object.

#### Returns

The error message.

#### ◆ **ob\_error\_get\_function()**

**OB\_EXPORT** const char \* ob\_error\_get\_function ( const **ob\_error** \* error )

Get the name of the API function that caused the error.

#### Parameters

[in] **error** The error object.

#### Returns

The name of the API function.

#### ◆ **ob\_error\_get\_args()**

**OB\_EXPORT** const char \* ob\_error\_get\_args ( const **ob\_error** \* error )

Get the error parameters.

#### Parameters

[in] **error** The error object.

#### Returns

The error parameters.

#### ◆ **ob\_error\_get\_exception\_type()**

**OB\_EXPORT** **ob\_exception\_type** ob\_error\_get\_exception\_type ( const **ob\_error** \* error )

Get the type of exception that caused the error.

#### Parameters

[in] **error** The error object.

#### Returns

The type of exception.

### ◆ **ob\_delete\_error()**

**OB\_EXPORT** void ob\_delete\_error ( **ob\_error** \* error )

Delete the error object.

#### Parameters

[in] **error** The error object to delete, you should set the pointer to NULL after calling this function.

Referenced by **ob::Error::handle()**, and **ob::Error::~Error()**.

# Error.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #ifndef __cplusplus
11 extern "C" {
12 #endif
13
14 #include "ObTypes.h"
15
26 OB_EXPORT ob_error *ob_create_error(ob_status status, const char *message, const char *function, con
    st char *args, ob_exception_type exception_type);
27
34 OB_EXPORT ob_status ob_error_get_status(const ob_error *error);
35
42 OB_EXPORT const char *ob_error_get_message(const ob_error *error);
43
50 OB_EXPORT const char *ob_error_get_function(const ob_error *error);
51
58 OB_EXPORT const char *ob_error_get_args(const ob_error *error);
59
66 OB_EXPORT ob_exception_type ob_error_get_exception_type(const ob_error *error);
67
73 OB_EXPORT void ob_delete_error(ob_error *error);
74
75 // The following interfaces are deprecated and are retained here for compatibility purposes.
76 #define ob_error_status ob_error_get_status
77 #define ob_error_message ob_error_get_message
78 #define ob_error_function ob_error_get_function
79 #define ob_error_args ob_error_get_args
80 #define ob_error_exception_type ob_error_get_exception_type
81
82 #ifndef __cplusplus
83 }
84 #endif
```

# Error.hpp File Reference

This file defines the Error class, which describes abnormal errors within the SDK. Detailed information about the exception can be obtained through this class. [More...](#)

```
#include "Types.hpp"
#include "libobsensor/h/Error.h"
#include <memory>
```

[Go to the source code of this file.](#)

## Classes

```
class ob::Error
```

## Namespaces

```
namespace ob
```

## Detailed Description

This file defines the Error class, which describes abnormal errors within the SDK. Detailed information about the exception can be obtained through this class.

Definition in file [Error.hpp](#).

# Error.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6
7
8
9
10
11 #include "Types.hpp"
12 #include "libobsensor/h/Error.h"
13 #include <memory>
14
15 namespace ob {
16     class Error : public std::exception {
17     private:
18         ob_error *impl_;
19
20         explicit Error(ob_error *error) : impl_(error) {};
21
22         Error& operator=(const Error&) = default;
23
24     public:
25         static void handle(ob_error **error, bool throw_exception = true) {
26             if(!error || !*error) { // no error
27                 return;
28             }
29
30             if(throw_exception) {
31                 throw Error(*error);
32             }
33             else {
34                 ob_delete_error(*error);
35                 *error = nullptr;
36             }
37         }
38
39         ~Error() override {
40             if(impl_) {
41                 ob_delete_error(impl_);
42                 impl_ = nullptr;
43             }
44         }
45
46         const char *what() const noexcept override {
47             return impl_->message;
48         }
49
50         OBExceptionType getExceptionType() const noexcept {
51             return impl_->exception_type;
52         }
53
54         const char *getFunction() const noexcept {
55             return impl_->function;
56         }
57
58         const char *getArgs() const noexcept {
59             return impl_->args;
60         }
61
62         const char *getMessage() const noexcept {
63             return impl_->message;
64         }
65
66         const char *getRawMessage() const noexcept {
67             return impl_->raw_message;
68         }
69
70         const char *getRawFunction() const noexcept {
71             return impl_->raw_function;
72         }
73
74         const char *getRawArgs() const noexcept {
75             return impl_->raw_args;
76         }
77
78         const char *getRawMessageWithFunction() const noexcept {
79             return impl_->raw_message_with_function;
80         }
81
82         const char *getRawMessageWithFunctionAndArgs() const noexcept {
83             return impl_->raw_message_with_function_and_args;
84         }
85
86         const char *getRawFunctionWithArgs() const noexcept {
87             return impl_->raw_function_with_args;
88         }
89
90         const char *getRawFunctionWithArgsAndMessage() const noexcept {
91             return impl_->raw_function_with_args_and_message;
92         }
93
94         const char *getRawFunctionWithMessage() const noexcept {
95             return impl_->raw_function_with_message;
96         }
97
98         const char *getRawFunctionWithMessageAndArgs() const noexcept {
99             return impl_->raw_function_with_message_and_args;
100
101         const char *getRawFunctionWithMessageAndFunction() const noexcept {
102             return impl_->raw_function_with_message_and_function;
103
104         const char *getRawFunctionWithMessageAndFunctionAndArgs() const noexcept {
105             return impl_->raw_function_with_message_and_function_and_args;
106         }
```

```
107     }
108
109 public:
110     // The following interfaces are deprecated and are retained here for compatibility purposes.
111     const char *getName() const noexcept {
112         return impl_->function;
113     }
114 };
115 } // namespace ob
116
```

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Export.h File Reference

Go to the source code of this file.

## Macros

```
#define OB_EXPORT __declspec(dllexport)  
#define OB_NO_EXPORT  
#define OB_DEPRECATED __declspec(deprecated)  
#define OB_DEPRECATED_EXPORT OB_EXPORT OB_DEPRECATED  
#define OB_DEPRECATED_NO_EXPORT OB_NO_EXPORT OB_DEPRECATED
```

## Macro Definition Documentation

### ◆ OB\_EXPORT

```
#define OB_EXPORT __declspec(dllexport)
```

Definition at line [15](#) of file [Export.h](#).

Referenced by [ob\\_accel\\_frame\\_get\\_temperature\(\)](#), [ob\\_accel\\_frame\\_get\\_value\(\)](#), [ob\\_accel\\_range\\_type\\_to\\_string\(\)](#), [ob\\_accel\\_stream\\_profile\\_get\\_full\\_scale\\_range\(\)](#), [ob\\_accel\\_stream\\_profile\\_get\\_intrinsic\(\)](#), [ob\\_accel\\_stream\\_profile\\_get\\_sample\\_rate\(\)](#), [ob\\_accel\\_stream\\_profile\\_set\\_intrinsic\(\)](#), [ob\\_align\\_filter\\_set\\_align\\_to\\_stream\\_profile\(\)](#), [ob\\_calibration\\_2d\\_to\\_2d\(\)](#), [ob\\_calibration\\_2d\\_to\\_3d\(\)](#), [ob\\_calibration\\_3d\\_to\\_2d\(\)](#), [ob\\_calibration\\_3d\\_to\\_3d\(\)](#), [ob\\_camera\\_param\\_list\\_get\\_count\(\)](#), [ob\\_camera\\_param\\_list\\_get\\_param\(\)](#), [ob\\_config\\_disable\\_all\\_stream\(\)](#), [ob\\_config\\_disable\\_stream\(\)](#), [ob\\_config\\_enable\\_accel\\_stream\(\)](#), [ob\\_config\\_enable\\_all\\_stream\(\)](#), [ob\\_config\\_enable\\_gyro\\_stream\(\)](#), [ob\\_config\\_enable\\_stream\(\)](#), [ob\\_config\\_enable\\_stream\\_with\\_stream\\_profile\(\)](#), [ob\\_config\\_enable\\_video\\_stream\(\)](#), [ob\\_config\\_get\\_enabled\\_stream\\_profile\\_list\(\)](#), [ob\\_config\\_set\\_align\\_mode\(\)](#), [ob\\_config\\_set\\_depth\\_scale\\_after\\_align\\_require\(\)](#), [ob\\_config\\_set\\_frame\\_aggregate\\_output\\_mode\(\)](#), [ob\\_create\\_accel\\_stream\\_profile\(\)](#), [ob\\_create\\_config\(\)](#), [ob\\_create\\_context\(\)](#), [ob\\_create\\_context\\_with\\_config\(\)](#), [ob\\_create\\_error\(\)](#), [ob\\_create\\_filter\(\)](#), [ob\\_create\\_frame\(\)](#), [ob\\_create\\_frame\\_from\\_buffer\(\)](#), [ob\\_create\\_frame\\_from\\_other\\_frame\(\)](#), [ob\\_create\\_frame\\_from\\_stream\\_profile\(\)](#), [ob\\_create\\_frameset\(\)](#), [ob\\_create\\_gyro\\_stream\\_profile\(\)](#), [ob\\_create\\_net\\_device\(\)](#), [ob\\_create\\_pipeline\(\)](#), [ob\\_create\\_pipeline\\_with\\_device\(\)](#), [ob\\_create\\_playback\\_device\(\)](#), [ob\\_create\\_private\\_filter\(\)](#), [ob\\_create\\_record\\_device\(\)](#), [ob\\_create\\_stream\\_profile\(\)](#), [ob\\_create\\_stream\\_profile\\_from\\_other\\_stream\\_profile\(\)](#), [ob\\_create\\_stream\\_profile\\_with\\_new\\_format\(\)](#), [ob\\_create\\_video\\_frame\(\)](#),

`ob_create_video_frame_from_buffer(), ob_create_video_stream_profile(),  
ob_delete_camera_param_list(), ob_delete_config(), ob_delete_context(),  
ob_delete_depth_work_mode_list(), ob_delete_device(), ob_delete_device_info(),  
ob_delete_device_list(), ob_delete_error(), ob_delete_filter(),  
ob_delete_filter_config_schema_list(), ob_delete_filter_list(), ob_delete_frame(),  
ob_delete_frame_interleave_list(), ob_delete_pipeline(), ob_delete_preset_list(),  
ob_delete_preset_resolution_config_list(), ob_delete_record_device(), ob_delete_sensor(),  
ob_delete_sensor_list(), ob_delete_stream_profile(), ob_delete_stream_profile_list(),  
ob_depth_frame_get_value_scale(), ob_depth_frame_set_value_scale(),  
ob_depth_work_mode_list_get_count(), ob_depth_work_mode_list_get_item(),  
ob_device_enable_global_timestamp(), ob_device_enable_heartbeat(),  
ob_device_export_current_settings_as_preset_json_data(),  
ob_device_export_current_settings_as_preset_json_file(),  
ob_device_frame_interleave_list_get_count(), ob_device_frame_interleave_list_get_name(),  
ob_device_frame_interleave_list_has_frame_interleave(),  
ob_device_get_available_frame_interleave_list(), ob_device_get_available_preset_list(),  
ob_device_get_available_preset_resolution_config_list(), ob_device_get_bool_property(),  
ob_device_get_bool_property_range(), ob_device_get_calibration_camera_param_list(),  
ob_device_get_current_depth_work_mode(), ob_device_get_current_depth_work_mode_name(),  
ob_device_get_current_preset_name(), ob_device_get_depth_work_mode_list(),  
ob_device_get_device_info(), ob_device_get_device_state(), ob_device_get_extension_info(),  
ob_device_get_float_property(), ob_device_get_float_property_range(),  
ob_device_get_int_property(), ob_device_get_int_property_range(),  
ob_device_get_multi_device_sync_config(), ob_device_get_raw_data(), ob_device_get_sensor(),  
ob_device_get_sensor_list(), ob_device_get_structured_data(),  
ob_device_get_supported_multi_device_sync_mode_bitmap(),  
ob_device_get_supported_property_count(), ob_device_get_supported_property_item(),  
ob_device_get_timestamp_reset_config(), ob_device_info_get_asicName(),  
ob_device_info_get_connection_type(), ob_device_info_get_device_type(),  
ob_device_info_get_firmware_version(), ob_device_info_get_hardware_version(),  
ob_device_info_get_ip_address(), ob_device_info_get_name(), ob_device_info_get_pid(),  
ob_device_info_get_serial_number(), ob_device_info_get_supported_min_sdk_version(),  
ob_device_info_get_uid(), ob_device_info_get_vid(), ob_device_is_extension_info_exist(),  
ob_device_is_frame_interleave_supported(), ob_device_is_global_timestamp_supported(),  
ob_device_is_property_supported(), ob_device_list_get_count(), ob_device_list_get_device(),  
ob_device_list_get_device_by_serial_number(), ob_device_list_get_device_by_uid(),  
ob_device_list_get_device_connection_type(), ob_device_list_get_device_ip_address(),  
ob_device_list_get_device_local_mac(), ob_device_list_get_device_name(),  
ob_device_list_get_device_pid(), ob_device_list_get_device_serial_number(),  
ob_device_list_get_device_uid(), ob_device_list_get_device_vid(),  
ob_device_load_frame_interleave(), ob_device_load_preset(),  
ob_device_load_preset_from_json_data(), ob_device_load_preset_from_json_file(),  
ob_device_preset_list_get_count(), ob_device_preset_list_get_name(),  
ob_device_preset_list_has_preset(), ob_device_preset_resolution_config_get_count(),`

`ob_device_preset_resolution_config_list_get_item(), ob_device_read_customer_data(),  
ob_device_reboot(), ob_device_send_and_receive_data(), ob_device_set_bool_property(),  
ob_device_set_float_property(), ob_device_set_int_property(),  
ob_device_set_multi_device_sync_config(), ob_device_set_state_changed_callback(),  
ob_device_set_structured_data(), ob_device_set_timestamp_reset_config(),  
ob_device_switch_depth_work_mode(), ob_device_switch_depth_work_mode_by_name(),  
ob_device_timer_sync_with_host(), ob_device_timestamp_reset(), ob_device_trigger_capture(),  
ob_device_update_firmware(), ob_device_update_firmware_from_data(),  
ob_device_update_optional_depth_presets(), ob_device_write_customer_data(),  
ob_disparity_based_stream_profile_get_disparity_param(),  
ob_disparity_based_stream_profile_set_disparity_param(), ob_enable_device_clock_sync(),  
ob_enable_net_device_enumeration(), ob_error_get_args(), ob_error_get_exception_type(),  
ob_error_get_function(), ob_error_get_message(), ob_error_get_status(),  
ob_filter_config_schema_list_get_count(), ob_filter_config_schema_list_get_item(),  
ob_filter_enable(), ob_filter_get_config_schema(), ob_filter_get_config_schema_list(),  
ob_filter_get_config_value(), ob_filter_get_name(), ob_filter_get_vendor_specific_code(),  
ob_filter_is_enabled(), ob_filter_list_get_count(), ob_filter_list_get_filter(), ob_filter_process(),  
ob_filter_push_frame(), ob_filter_reset(), ob_filter_set_callback(), ob_filter_set_config_value(),  
ob_filter_update_config(), ob_format_to_string(), ob_format_type_to_string(),  
ob_frame_add_ref(), ob_frame_copy_info(), ob_frame_get_data(), ob_frame_get_data_size(),  
ob_frame_get_device(), ob_frame_get_format(), ob_frame_get_global_timestamp_us(),  
ob_frame_get_index(), ob_frame_get_metadata(), ob_frame_get_metadata_size(),  
ob_frame_get_metadata_value(), ob_frame_get_sensor(), ob_frame_get_stream_profile(),  
ob_frame_get_system_timestamp_us(), ob_frame_get_timestamp_us(), ob_frame_get_type(),  
ob_frame_has_metadata(), ob_frame_set_stream_profile(),  
ob_frame_set_system_timestamp_us(), ob_frame_set_timestamp_us(), ob_frame_type_to_string(),  
ob_frame_update_data(), ob_frame_update_metadata(), ob_frameset_get_color_frame(),  
ob_frameset_get_count(), ob_frameset_get_depth_frame(), ob_frameset_get_frame(),  
ob_frameset_get_frame_by_index(), ob_frameset_get_ir_frame(),  
ob_frameset_get_points_frame(), ob_frameset_push_frame(), ob_free_idle_memory(),  
ob_get_d2c_depth_profile_list(), ob_get_major_version(), ob_get_minor_version(),  
ob_get_patch_version(), ob_get_stage_version(), ob_get_version(),  
ob_gyro_frame_get_temperature(), ob_gyro_frame_get_value(), ob_gyro_range_type_to_string(),  
ob_gyro_stream_get_intrinsic(), ob_gyro_stream_profile_get_full_scale_range(),  
ob_gyro_stream_profile_get_sample_rate(), ob_gyro_stream_set_intrinsic(),  
ob_imu_rate_type_to_string(), ob_ir_frame_get_source_sensor_type(),  
ob_meta_data_type_to_string(), ob_pipeline_disable_frame_sync(),  
ob_pipeline_enable_frame_sync(), ob_pipeline_get_calibration_param(),  
ob_pipeline_get_camera_param(), ob_pipeline_get_camera_param_with_profile(),  
ob_pipeline_get_config(), ob_pipeline_get_device(), ob_pipeline_get_stream_profile_list(),  
ob_pipeline_start(), ob_pipeline_start_with_callback(), ob_pipeline_start_with_config(),  
ob_pipeline_stop(), ob_pipeline_switch_config(), ob_pipeline_wait_for_frameset(),  
ob_playback_device_get_current_playback_status(), ob_playback_device_get_duration(),  
ob_playback_device_get_position(), ob_playback_device_pause(), ob_playback_device_resume(),`

`ob_playback_device_seek(), ob_playback_device_set_playback_rate(),  
ob_playback_device_set_playback_status_changed_callback(), ob_point_cloud_frame_get_height(),  
ob_point_cloud_frame_get_width(), ob_points_frame_get_coordinate_value_scale(),  
ob_query_device_list(), ob_record_device_pause(), ob_record_device_resume(),  
ob_save_pointcloud_to_ply(), ob_sensor_create_recommended_filter_list(),  
ob_sensor_get_stream_profile_list(), ob_sensor_get_type(), ob_sensor_list_get_count(),  
ob_sensor_list_get_sensor(), ob_sensor_list_get_sensor_by_type(),  
ob_sensor_list_get_sensor_type(), ob_sensor_start(), ob_sensor_stop(), ob_sensor_switch_profile(),  
ob_sensor_type_to_stream_type(), ob_sensor_type_to_string(), ob_set_device_changed_callback(),  
ob_set_extensions_directory(), ob_set_logger_severity(), ob_set_logger_to_callback(),  
ob_set_logger_to_console(), ob_set_logger_to_file(), ob_set_uvc_backend_type(),  
ob_stream_profile_get_extrinsic_to(), ob_stream_profile_get_format(),  
ob_stream_profile_get_type(), ob_stream_profile_list_get_accel_stream_profile(),  
ob_stream_profile_list_get_count(), ob_stream_profile_list_get_gyro_stream_profile(),  
ob_stream_profile_list_get_profile(), ob_stream_profile_list_get_video_stream_profile(),  
ob_stream_profile_set_extrinsic_to(), ob_stream_profile_set_extrinsic_to_type(),  
ob_stream_profile_set_format(), ob_stream_profile_set_type(), ob_stream_type_to_string(),  
ob_transformation_2d_to_2d(), ob_transformation_2d_to_3d(), ob_transformation_3d_to_2d(),  
ob_transformation_3d_to_3d(), ob_video_frame_get_height(),  
ob_video_frame_get_pixel_available_bit_size(), ob_video_frame_get_pixel_type(),  
ob_video_frame_get_width(), ob_video_frame_set_pixel_available_bit_size(),  
ob_video_frame_set_pixel_type(), ob_video_stream_profile_get_distortion(),  
ob_video_stream_profile_get_fps(), ob_video_stream_profile_get_height(),  
ob_video_stream_profile_get_intrinsic(), ob_video_stream_profile_get_width(),  
ob_video_stream_profile_set_distortion(), ob_video_stream_profile_set_height(),  
ob_video_stream_profile_set_intrinsic(), ob_video_stream_profile_set_width(),  
transformation_depth_frame_to_color_camera(), transformation_depth_to_pointcloud(),  
transformation_depth_to_rgbd_pointcloud(), and transformation_init_xy_tables().`

#### ◆ OB\_NO\_EXPORT

```
#define OB_NO_EXPORT
```

Definition at line **20** of file [Export.h](#).

#### ◆ OB\_DEPRECATED

```
#define OB_DEPRECATED __declspec(deprecated)
```

Definition at line **25** of file [Export.h](#).

◆ OB\_DEPRECATED\_EXPORT

```
#define OB_DEPRECATED_EXPORT OB_EXPORT OB_DEPRECATED
```

Definition at line **29** of file [Export.h](#).

◆ OB\_DEPRECATED\_NO\_EXPORT

```
#define OB_DEPRECATED_NO_EXPORT OB_NO_EXPORT OB_DEPRECATED
```

Definition at line **33** of file [Export.h](#).

# Export.h

Go to the documentation of this file.

```
1  ifndef OB_EXPORT_H
2  define OB_EXPORT_H
3
4
5  ifdef OB_STATIC_DEFINE
6  # define OB_EXPORT
7  # define OB_NO_EXPORT
8  #else
9  # ifndef OB_EXPORT
10 #  ifdef OrbbecSDK_EXPORTS
11     /* We are building this library */
12 #   define OB_EXPORT __declspec(dllexport)
13 #  else
14     /* We are using this library */
15 #   define OB_EXPORT __declspec(dllimport)
16 #  endif
17 # endif
18
19 # ifndef OB_NO_EXPORT
20 #  define OB_NO_EXPORT
21 # endif
22 #endif
23
24 #ifndef OB_DEPRECATED
25 #  define OB_DEPRECATED __declspec(deprecated)
26 #endif
27
28 #ifndef OB_DEPRECATED_EXPORT
29 #  define OB_DEPRECATED_EXPORT OB_EXPORT OB_DEPRECATED
30 #endif
31
32 #ifndef OB_DEPRECATED_NO_EXPORT
33 #  define OB_DEPRECATED_NO_EXPORT OB_NO_EXPORT OB_DEPRECATED
34 #endif
35
36 /* NOLINTNEXTLINE(readability-avoid-unconditional-preprocessor-if) */
37 #if 0 /* DEFINE_NO_DEPRECATED */
38 #  ifndef OB_NO_DEPRECATED
39 #   define OB_NO_DEPRECATED
40 #  endif
41 #endif
42
43 #endif/* OB_EXPORT_H */
```

# Filter.h File Reference

The processing unit of the SDK can perform point cloud generation, format conversion and other functions.

[More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_get_filter ob_filter_list_get_filter  
#define ob_get_filter_name ob_filter_get_name
```

## Functions

**OB\_EXPORT** **ob\_filter** \* **ob\_create\_filter** (const char \*name, **ob\_error** \*\*error)  
Create a Filter object.

**OB\_EXPORT** const char \* **ob\_filter\_get\_name** (const **ob\_filter** \*filter, **ob\_error** \*\*error)  
Get the name of **ob\_filter**.

**OB\_EXPORT** const char \* **ob\_filter\_get\_vendor\_specific\_code** (const char \*name, **ob\_error** \*\*error)  
Get the vendor specific code of a filter by filter name.

**OB\_EXPORT** **ob\_filter** \* **ob\_create\_private\_filter** (const char \*name, const char \*activation\_key, **ob\_error** \*\*error)  
Create a private Filter object with activation key.

**OB\_EXPORT** void **ob\_delete\_filter** (**ob\_filter** \*filter, **ob\_error** \*\*error)  
Delete the filter.

**OB\_EXPORT** const char \* **ob\_filter\_get\_config\_schema** (const **ob\_filter** \*filter, **ob\_error** \*\*error)  
Get config schema of the filter.

**OB\_EXPORT** **ob\_filter\_config\_schema\_list** \* **ob\_filter\_get\_config\_schema\_list** (const **ob\_filter** \*filter, **ob\_error** \*\*error)  
Get the filter config schema list of the filter.

**OB\_EXPORT** void **ob\_delete\_filter\_config\_schema\_list**  
(**ob\_filter\_config\_schema\_list** \*config\_schema\_list, **ob\_error** \*\*error)  
Delete a list of filter config schema items.

**OB\_EXPORT** void **ob\_filter\_update\_config** (**ob\_filter** \*filter, uint8\_t argc,  
const char \*\*argv, **ob\_error** \*\*error)

Update config of the filter.

**OB\_EXPORT** double **ob\_filter\_get\_config\_value** (const **ob\_filter** \*filter,  
const char \*config\_name, **ob\_error** \*\*error)

Get the filter config value by name and cast to double.

**OB\_EXPORT** void **ob\_filter\_set\_config\_value** (**ob\_filter** \*filter, const char  
\*config\_name, double value, **ob\_error** \*\*error)

Set the filter config value by name.

**OB\_EXPORT** void **ob\_filter\_reset** (**ob\_filter** \*filter, **ob\_error** \*\*error)

Reset the filter, clears the cache, and resets the state. If  
the asynchronous interface is used, the processing  
thread will also be stopped and the pending cache  
frames will be cleared.

**OB\_EXPORT** void **ob\_filter\_enable** (**ob\_filter** \*filter, bool enable,  
**ob\_error** \*\*error)

Enable the frame post processing.

**OB\_EXPORT** bool **ob\_filter\_is\_enabled** (const **ob\_filter** \*filter, **ob\_error**  
\*\*error)

Get the enable status of the frame post processing.

**OB\_EXPORT** **ob\_frame** \* **ob\_filter\_process** (**ob\_filter** \*filter, const **ob\_frame**  
\*frame, **ob\_error** \*\*error)

Process the frame (synchronous interface).

**OB\_EXPORT** void **ob\_filter\_set\_callback** (**ob\_filter** \*filter,  
**ob\_filter\_callback** callback, void \*user\_data, **ob\_error**  
\*\*error)

Set the processing result callback function for the filter  
(asynchronous callback interface).

**OB\_EXPORT** void **ob\_filter\_push\_frame** (**ob\_filter** \*filter, const  
**ob\_frame** \*frame, **ob\_error** \*\*error)

Push the frame into the pending cache for the filter  
(asynchronous callback interface).

**OB\_EXPORT** uint32\_t **ob\_filter\_list\_get\_count** (const **ob\_filter\_list**  
\*filter\_list, **ob\_error** \*\*error)

Get the number of filter in the list.

**OB\_EXPORT** **ob\_filter** \* **ob\_filter\_list\_get\_filter** (const **ob\_filter\_list**  
\*filter\_list, uint32\_t index, **ob\_error** \*\*error)

Get the filter by index.

**OB\_EXPORT** void **ob\_delete\_filter\_list** (**ob\_filter\_list** \*filter\_list,  
**ob\_error** \*\*error)

Delete a list of **ob\_filter** objects.

**OB\_EXPORT** uint32\_t **ob\_filter\_config\_schema\_list\_get\_count** (const  
**ob\_filter\_config\_schema\_list** \*config\_schema\_list,  
**ob\_error** \*\*error)

Get the number of config schema items in the config schema list.

**OB\_EXPORT** **ob\_filter\_config\_schema\_item** **ob\_filter\_config\_schema\_list\_get\_item** (const  
**ob\_filter\_config\_schema\_list** \*config\_schema\_list,  
uint32\_t index, **ob\_error** \*\*error)

Get the config schema item by index.

**OB\_EXPORT** void **ob\_align\_filter\_set\_align\_to\_stream\_profile**  
(**ob\_filter** \*filter, const **ob\_stream\_profile**  
\*align\_to\_stream\_profile, **ob\_error** \*\*error)

Set the align to stream profile for the align filter. @breif  
It is useful when the align target stream dose not started  
(without any frame to get intrinsics and extrinsics).

## Detailed Description

The processing unit of the SDK can perform point cloud generation, format conversion and other functions.

Definition in file **Filter.h**.

## Macro Definition Documentation

### ◆ ob\_get\_filter

```
#define ob_get_filter ob_filter_list_get_filter
```

Definition at line **259** of file **Filter.h**.

### ◆ ob\_get\_filter\_name

```
#define ob_get_filter_name ob_filter_get_name
```

Definition at line **260** of file **Filter.h**.

## Function Documentation

### ◆ ob\_create\_filter()

```
OB_EXPORT ob_filter * ob_create_filter ( const char * name,  
                                         ob_error ** error )
```

Create a Filter object.

#### Attention

If the filter of the specified name is a private filter, and the creator of the filter have not been activated, the function will return NULL.

#### Parameters

**name** The name of the filter.

**error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Align::Align\(\)](#), [ob::FilterFactory::createFilter\(\)](#),  
[ob::DecimationFilter::DecimationFilter\(\)](#), [ob::FormatConvertFilter::FormatConvertFilter\(\)](#),  
[ob::HdrMerge::HdrMerge\(\)](#), [ob::PointCloudFilter::PointCloudFilter\(\)](#),  
[ob::SequenceIdFilter::SequenceIdFilter\(\)](#), and [ob::ThresholdFilter::ThresholdFilter\(\)](#).

### ◆ ob\_filter\_get\_name()

```
OB_EXPORT const char * ob_filter_get_name ( const ob_filter * filter,  
                                         ob_error ** error )
```

Get the name of **ob\_filter**.

#### Parameters

**filter** **ob\_filter** object

**error** Pointer to an error object that will be set if an error occurs.

#### Returns

char The filter of name

Referenced by [ob::Filter::init\(\)](#).

### ◆ ob\_filter\_get\_vendor\_specific\_code()

Get the vendor specific code of a filter by filter name.

A private filter can define its own vendor specific code for specific purposes.

## Parameters

**name** The name of the filter.

**error** Pointer to an error object that will be set if an error occurs.

## Returns

const char\* Return the vendor specific code of the filter.

Referenced by [ob::FilterFactory::getFilterVendorSpecificCode\(\)](#).

- #### ◆ ob\_create\_private\_filter()

Create a private Filter object with activation key.

Some private filters require an activation key to be activated, its depends on the vendor of the filter.

## Parameters

**name** The name of the filter.

**activation\_key** The activation key of the filter.

**error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob filter\*** Return the private filter object.

Referenced by [ob::FilterFactory::createPrivateFilter\(\)](#),  
[ob::DisparityTransform::DisparityTransform\(\)](#), [ob::HoleFillingFilter::HoleFillingFilter\(\)](#),  
[ob::NoiseRemovalFilter::NoiseRemovalFilter\(\)](#),  
[ob::SpatialAdvancedFilter::SpatialAdvancedFilter\(\)](#), and [ob::TemporalFilter::TemporalFilter\(\)](#).

- #### ◆ ob delete filter()

```
OB_EXPORT void ob_delete_filter ( ob_filter * filter,  
                                ob_error ** error )
```

Delete the filter.

### Parameters

- [in] **filter** The filter object to be deleted.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Filter::~Filter\(\)](#).

### ◆ [ob\\_filter\\_get\\_config\\_schema\(\)](#)

```
OB_EXPORT const char * ob_filter_get_config_schema ( const ob_filter * filter,  
                                                    ob_error ** error )
```

Get config schema of the filter.

The returned string is a csv format string representing the configuration schema of the filter. The format of the string is: <parameter\_name>, <parameter\_type: "int", "float", "bool">, <minimum\_value>, <maximum\_value>, <value\_step>, <default\_value>, <parameter\_description>

### Parameters

- [in] **filter** The filter object to get the configuration schema for
- [out] **error** Pointer to an error object that will be set if an error occurs

### Returns

A csv format string representing the configuration schema of the filter

Referenced by [ob::Filter::getConfigSchema\(\)](#).

### ◆ [ob\\_filter\\_get\\_config\\_schema\\_list\(\)](#)

**OB\_EXPORT ob\_filter\_config\_schema\_list**\* ob\_filter\_get\_config\_schema\_list ( const **ob\_filter** \* filter,  
**ob\_error** \*\* error )

Get the filter config schema list of the filter.

The returned string is a list of `ob_config_schema_item` representing the configuration schema of the filter.

## Attention

The returned list should be deleted by calling `ob_delete_filter_config_schema_list` when it is no longer needed.

## Parameters

**filter** The filter object to get the configuration schema for

**error** Pointer to an error object that will be set if an error occurs

## Returns

ob filter config schema list\* Return the filter config schema list of the filter

Referenced by **ob::Filter::init()**.

- ◆ ob delete filter config schema list()

Delete a list of filter config schema items.

## Parameters

**config\_schema\_list** The list of filter config schema items to delete.

**error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Filter::init()**.

- ## ◆ ob\_filter\_update\_config()

```
OB_EXPORT void ob_filter_update_config ( ob_filter * filter,  
                                         uint8_t argc,  
                                         const char ** argv,  
                                         ob_error ** error )
```

Update config of the filter.

### Attention

The passed in argc and argv must match the configuration schema returned by the [ob\\_filter\\_get\\_config\\_schema](#) function.

### Parameters

- [in] **filter** The filter object to update the configuration for
- [in] **argc** The number of arguments in the argv array
- [in] **argv** An array of strings representing the configuration values
- [out] **error** Pointer to an error object that will be set if an error occurs

### ◆ [ob\\_filter\\_get\\_config\\_value\(\)](#)

```
OB_EXPORT double ob_filter_get_config_value ( const ob_filter * filter,  
                                              const char * config_name,  
                                              ob_error ** error )
```

Get the filter config value by name and cast to double.

### Attention

The returned value is cast to double, the actual type of the value depends on the filter config schema returned by [ob\\_filter\\_get\\_config\\_schema](#).

### Parameters

- [in] **filter** A filter object.
- [in] **config\_name** config name
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

double The value of the config.

Referenced by [ob::Filter::getConfigValue\(\)](#).

◆ [ob\\_filter\\_set\\_config\\_value\(\)](#)

```
OB_EXPORT void ob_filter_set_config_value ( ob_filter * filter,  
                                         const char * config_name,  
                                         double value,  
                                         ob_error ** error )
```

Set the filter config value by name.

**Attention**

The pass into value type is double, which will be cast to the actual type inside the filter. The actual type can be queried by the filter config schema returned by [ob\\_filter\\_get\\_config\\_schema](#).

**Parameters**

[in] **filter** A filter object.  
[in] **config\_name** config name  
[in] **value** The value to set.  
[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Filter::setConfigValue\(\)](#).

◆ [ob\\_filter\\_reset\(\)](#)

```
OB_EXPORT void ob_filter_reset ( ob_filter * filter,  
                                 ob_error ** error )
```

Reset the filter, clears the cache, and resets the state. If the asynchronous interface is used, the processing thread will also be stopped and the pending cache frames will be cleared.

**Parameters**

[in] **filter** A filter object.  
[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Filter::reset\(\)](#).

◆ [ob\\_filter\\_enable\(\)](#)

```
OB_EXPORT void ob_filter_enable ( ob_filter * filter,  
                                bool enable,  
                                ob_error ** error )
```

Enable the frame post processing.

The filter default is enable.

### Attention

If the filter has been disabled by calling this function, processing will directly output a clone of the input frame.

### Parameters

[in] **filter** A filter object.

[in] **enable** enable status, true: enable; false: disable.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Filter::enable\(\)](#).

### ◆ ob\_filter\_is\_enabled()

```
OB_EXPORT bool ob_filter_is_enabled ( const ob_filter * filter,  
                                     ob_error ** error )
```

Get the enable status of the frame post processing.

### Attention

If the filter is disabled, the processing will directly output a clone of the input frame.

### Parameters

[in] **filter** A filter object.

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The post processing filter status. True: enable; False: disable.

Referenced by [ob::Filter::isEnabled\(\)](#).

### ◆ ob\_filter\_process()

```
OB_EXPORT ob_frame * ob_filter_process ( ob_filter * filter,  
                                         const ob_frame * frame,  
                                         ob_error ** error )
```

Process the frame (synchronous interface).

### Parameters

- [in] **filter** A filter object.
- [in] **frame** Pointer to the frame object to be processed.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The frame object processed by the filter.

Referenced by [ob::Filter::process\(\)](#).

## ◆ [ob\\_filter\\_set\\_callback\(\)](#)

```
OB_EXPORT void ob_filter_set_callback ( ob_filter * filter,  
                                         ob_filter_callback callback,  
                                         void * user_data,  
                                         ob_error ** error )
```

Set the processing result callback function for the filter (asynchronous callback interface).

### Parameters

- [in] **filter** A filter object.
- [in] **callback** Callback function.
- [in] **user\_data** Arbitrary user data pointer can be passed in and returned from the callback.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Filter::setCallBack\(\)](#).

## ◆ [ob\\_filter\\_push\\_frame\(\)](#)

```
OB_EXPORT void ob_filter_push_frame ( ob_filter * filter,
                                         const ob_frame * frame,
                                         ob_error ** error )
```

Push the frame into the pending cache for the filter (asynchronous callback interface).

The frame will be processed by the filter when the processing thread is available and return a new processed frame to the callback function.

### Attention

The frame object will be add reference count, so the user still need call **ob\_delete\_frame** to release the frame after calling this function.

### Parameters

- [in] **filter** A filter object.
- [in] **frame** Pointer to the frame object to be processed.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Filter::pushFrame\(\)](#).

### ◆ [ob\\_filter\\_list\\_get\\_count\(\)](#)

```
OB_EXPORT uint32_t ob_filter_list_get_count ( const ob_filter_list * filter_list,
                                              ob_error ** error )
```

Get the number of filter in the list.

### Parameters

- [in] **filter\_list** filter list
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**uint32\_t** The number of list

Referenced by [ob::Sensor::createRecommendedFilters\(\)](#), and [ob::OBFilterList::getCount\(\)](#).

### ◆ [ob\\_filter\\_list\\_get\\_filter\(\)](#)

```
OB_EXPORT ob_filter * ob_filter_list_get_filter ( const ob_filter_list * filter_list,  
                                              uint32_t index,  
                                              ob_error ** error )
```

Get the filter by index.

#### Parameters

- [in] **filter\_list** Filter list
- [in] **index** Filter index
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_filter** The index of **ob\_filter**

Referenced by **ob::Sensor::createRecommendedFilters()**, and **ob::OBFilterList::getFilter()**.

#### ◆ **ob\_delete\_filter\_list()**

```
OB_EXPORT void ob_delete_filter_list ( ob_filter_list * filter_list,  
                                         ob_error ** error )
```

Delete a list of **ob\_filter** objects.

#### Parameters

- [in] **filter\_list** The list of **ob\_filter** objects to delete.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Sensor::createRecommendedFilters()**, and **ob::OBFilterList::~OBFilterList()**.

#### ◆ **ob\_filter\_config\_schema\_list\_get\_count()**

```
OB_EXPORT uint32_t
ob_filter_config_schema_list_get_count
                                ( const ob_filter_config_schema_list * config_schema_list,
                                  ob_error ** error )
```

Get the number of config schema items in the config schema list.

#### Parameters

**config\_schema\_list** Filter config schema list  
**error** Pointer to an error object that will be set if an error occurs.

#### Returns

**uint32\_t** The number of config schema items in the filter list

Referenced by **ob::Filter::init()**.

◆ **ob\_filter\_config\_schema\_list\_get\_item()**

**OB\_EXPORT ob\_filter\_config\_schema\_item**

```
ob_filter_config_schema_list_get_item
                                ( const ob_filter_config_schema_list * config_schema_list,
                                  uint32_t index,
                                  ob_error ** error )
```

Get the config schema item by index.

#### Parameters

**config\_schema\_list** Filter config schema list  
**index** Config schema item index  
**error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_filter\_config\_schema\_item\*** The config schema item by index

Referenced by **ob::Filter::init()**.

◆ **ob\_align\_filter\_set\_align\_to\_stream\_profile()**

```
OB_EXPORT void
ob_align_filter_set_align_to_stream_profile
( ob_filter * filter,
  const ob_stream_profile * align_to_stream_profile,
  ob_error ** error )
```

Set the align to stream profile for the align filter. @breif It is useful when the align target stream dose not started (without any frame to get intrinsics and extrinsics).

## Parameters

**filter**

A filter object.

**align\_to\_stream\_profile**

The align target stream profile.

**error**

Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Align::setAlignToStreamProfile\(\)](#).

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Filter.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #ifndef __cplusplus
11 extern "C" {
12 #endif
13
14 #include "ObTypes.h"
15
24 OB_EXPORT ob_filter *ob_create_filter(const char *name, ob_error **error);
25
33 OB_EXPORT const char *ob_filter_get_name(const ob_filter *filter, ob_error **error);
34
43 OB_EXPORT const char *ob_filter_get_vendor_specific_code(const char *name, ob_error **error);
44
55 OB_EXPORT ob_filter *ob_create_private_filter(const char *name, const char *activation_key, ob_error *error);
56
63 OB_EXPORT void ob_delete_filter(ob_filter *filter, ob_error **error);
64
75 OB_EXPORT const char *ob_filter_get_config_schema(const ob_filter *filter, ob_error **error);
76
87 OB_EXPORT ob_filter_config_schema_list *ob_filter_get_config_schema_list(const ob_filter *filter, ob_error **error);
88
95 OB_EXPORT void ob_delete_filter_config_schema_list(ob_filter_config_schema_list *config_schema_list, ob_error **error);
96
107 OB_EXPORT void ob_filter_update_config(ob_filter *filter, uint8_t argc, const char **argv, ob_error **error);
108
120 OB_EXPORT double ob_filter_get_config_value(const ob_filter *filter, const char *config_name, ob_error **error);
121
133 OB_EXPORT void ob_filter_set_config_value(ob_filter *filter, const char *config_name, double value, ob_error **error);
134
142 OB_EXPORT void ob_filter_reset(ob_filter *filter, ob_error **error);
143
154 OB_EXPORT void ob_filter_enable(ob_filter *filter, bool enable, ob_error **error);
155
166 OB_EXPORT bool ob_filter_is_enabled(const ob_filter *filter, ob_error **error);
167
177 OB_EXPORT ob_frame *ob_filter_process(ob_filter *filter, const ob_frame *frame, ob_error **error);
178
187 OB_EXPORT void ob_filter_set_callback(ob_filter *filter, ob_filter_callback callback, void *user_data, ob_error **error);
188
199 OB_EXPORT void ob_filter_push_frame(ob_filter *filter, const ob_frame *frame, ob_error **error);
200
208 OB_EXPORT uint32_t ob_filter_list_get_count(const ob_filter_list *filter_list, ob_error **error);
209
218 OB_EXPORT ob_filter *ob_filter_list_get_filter(const ob_filter_list *filter_list, uint32_t index, ob_error **error);
219
226 OB_EXPORT void ob_delete_filter_list(ob_filter_list *filter_list, ob_error **error);
```

```
227
235 OB_EXPORT uint32_t ob_filter_config_schema_list_get_count(const ob_filter_config_schema_list *conf
236 ig_schema_list, ob_error **error);
245 OB_EXPORT ob_filter_config_schema_item ob_filter_config_schema_list_get_item(const ob_filter_config_
246 schema_list *config_schema_list, uint32_t index,
247 ob_error **error);
256 OB_EXPORT void ob_align_filter_set_align_to_stream_profile(ob_filter *filter, const ob_stream_profile *
257 align_to_stream_profile, ob_error **error);
258 // The following interfaces are deprecated and are retained here for compatibility purposes.
259 #define ob_get_filter ob_filter_list_get_filter
260 #define ob_get_filter_name ob_filter_get_name
261
262 #ifdef __cplusplus
263 }
264 #endif
265
```

# Filter.hpp File Reference

This file contains the Filter class, which is the processing unit of the SDK that can perform point cloud generation, format conversion, and other functions. [More...](#)

```
#include "Types.hpp"
#include "Error.hpp"
#include "Frame.hpp"
#include "libobsensor/h/Filter.h"
#include "libobsensor/h/Frame.h"
#include <functional>
#include <memory>
#include <map>
#include <string>
#include <iostream>
#include <vector>
#include <typeinfo>
#include <typeindex>
#include <unordered_map>
#include <cstring>
```

[Go to the source code of this file.](#)

## Classes

struct **ob::RangeTraits< T >**

Get the type of a PropertyRange member. [More...](#)

struct **ob::RangeTraits< OB UInt8PropertyRange >**

struct **ob::Range Traits< OB UInt16PropertyRange >**

struct **ob::Range Traits< OB IntPropertyRange >**

struct **ob::Range Traits< OB FloatPropertyRange >**

class **ob::Filter**

The **Filter** class is the base class for all filters in the SDK. [More...](#)

class **ob::FilterFactory**

A factory class for creating filters. [More...](#)

class **ob::PointCloudFilter**

The **PointCloudFilter** class is a subclass of **Filter** that generates point clouds. [More...](#)

class **ob::Align**

**Align** for depth to other or other to depth. [More...](#)

class **ob::FormatConvertFilter**

The **FormatConvertFilter** class is a subclass of **Filter** that performs format conversion. [More...](#)

#### class **ob::HdrMerge**

**HdrMerge** processing block, the processing merges between depth frames with different sub-preset sequence ids. [More...](#)

#### class **ob::SequenceIdFilter**

Create **SequenceIdFilter** processing block. [More...](#)

#### class **ob::DecimationFilter**

Decimation filter, reducing complexity by subsampling depth maps and losing depth details. [More...](#)

#### class **ob::ThresholdFilter**

Creates depth Thresholding filter By controlling min and max options on the block. [More...](#)

#### class **ob::SpatialAdvancedFilter**

Spatial advanced filte smooths the image by calculating frame with alpha and delta settings alpha defines the weight of the current pixel for smoothing, delta defines the depth gradient below which the smoothing will occur as number of depth levels. [More...](#)

#### class **ob::HoleFillingFilter**

Hole filling filter, the processing performed depends on the selected hole filling mode. [More...](#)

#### class **ob::NoiseRemovalFilter**

The noise removal filter,removing scattering depth pixels. [More...](#)

#### class **ob::TemporalFilter**

Temporal filter. [More...](#)

#### class **ob::DisparityTransform**

Depth to disparity or disparity to depth. [More...](#)

#### class **ob::OBFilterList**

## Namespaces

namespace **ob**

## Typedefs

typedef std::function< void(std::shared\_ptr<**Frame** >) > **ob::FilterCallback**

A callback function that takes a shared pointer to a **Frame** object as its argument.

## Functions

template<typename T>

T **ob::getPropertyRange** (const **OBFilterConfigSchemaItem** &item, const double cur)

Get T Property Range.

## Detailed Description

This file contains the Filter class, which is the processing unit of the SDK that can perform point cloud generation, format conversion, and other functions.

Definition in file **Filter.hpp**.

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# Filter.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6
7 #include "Types.hpp"
8 #include "Error.hpp"
9 #include "Frame.hpp"
10 #include "libobsensor/h/Filter.h"
11 #include "libobsensor/h/Frame.h"
12 #include <functional>
13 #include <memory>
14 #include <map>
15 #include <string>
16 #include <iostream>
17 #include <vector>
18 #include <typeinfo>
19 #include <typeindex>
20 #include <unordered_map>
21 #include <cstring>
22
23 namespace ob {
24
25     typedef std::function<void(std::shared_ptr<Frame>)> FilterCallback;
26
27     template <typename T> struct RangeTraits {
28         using valueType = void;
29     };
30
31     template <> struct RangeTraits<OBUInt8PropertyRange> {
32         using valueType = uint8_t;
33     };
34
35     template <> struct RangeTraits<OBUInt16PropertyRange> {
36         using valueType = uint16_t;
37     };
38
39     template <> struct RangeTraits<OBIntPropertyRange> {
40         using valueType = uint32_t;
41     };
42
43     template <> struct RangeTraits<OBFloatPropertyRange> {
44         using valueType = float;
45     };
46
47     template <typename T> T getPropertyRange(const OBFilterConfigSchemaItem& item, const double cur) {
48         // If T type is illegal, T will be void
49         using valueType = typename RangeTraits<T>::valueType;
50         T range{};
51
52         // Compilate error will be reported here if T is void
53         range.cur = static_cast<valueType>(cur);
54         range.def = static_cast<valueType>(item.def);
55         range.max = static_cast<valueType>(item.max);
56         range.min = static_cast<valueType>(item.min);
57         range.step = static_cast<valueType>(item.step);
58
59         return range;
60     }
61 }
```

```

76 class Filter : public std::enable_shared_from_this<Filter> {
77 protected:
78     ob_filter *impl_ = nullptr;
79     std::string name_;
80     FilterCallback callback_;
81     std::vector<OBFilterConfigSchemaItem> configSchemaVec_;
82
83 protected:
84     Filter() = default;
85
86     virtual void init(ob_filter *impl) {
87         impl_ = impl;
88         ob_error *error = nullptr;
89         name_ = ob_filter_get_name(impl_, &error);
90         Error::handle(&error);
91
92         auto configSchemaList = ob_filter_get_config_schema_list(impl_, &error);
93         Error::handle(&error);
94
95         auto count = ob_filter_config_schema_list_get_count(configSchemaList, &error);
96         Error::handle(&error);
97
98         auto item = ob_filter_config_schema_list_get_item(configSchemaList, i, &error);
99         Error::handle(&error);
100        configSchemaVec_.emplace_back(item);
101    }
102
103    ob_delete_filter_config_schema_list(configSchemaList, &error);
104    Error::handle(&error);
105 }
106
107 public:
108     explicit Filter(ob_filter *impl) {
109         init(impl);
110     }
111
112     virtual ~Filter() noexcept {
113         if(impl_ != nullptr) {
114             ob_error *error = nullptr;
115             ob_delete_filter(impl_, &error);
116             Error::handle(&error, false);
117         }
118     }
119
120     ob_filter *getImpl() const {
121         return impl_;
122     }
123
124     virtual const std::string &getName() const {
125         return name_;
126     }
127
128     virtual void reset() const {
129         ob_error *error = nullptr;
130         ob_filter_reset(impl_, &error);
131         Error::handle(&error);
132     }
133
134     virtual void enable(bool enable) const {
135         ob_error *error = nullptr;
136         ob_filter_enable(impl_, enable, &error);
137         Error::handle(&error);
138     }
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160

```

```

164     virtual bool isEnabled() const {
165         ob_error *error = nullptr;
166         bool enable = ob_filter_is_enabled(impl_, &error);
167         Error::handle(&error);
168         return enable;
169     }
170
177     virtual std::shared_ptr<Frame> process(std::shared_ptr<const Frame> frame) const {
178         ob_error *error = nullptr;
179         auto result = ob_filter_process(impl_, frame->getImpl(), &error);
180         Error::handle(&error);
181         if(!result) {
182             return nullptr;
183         }
184         return std::make_shared<Frame>(result);
185     }
186
192     virtual void pushFrame(std::shared_ptr<Frame> frame) const {
193         ob_error *error = nullptr;
194         ob_filter_push_frame(impl_, frame->getImpl(), &error);
195         Error::handle(&error);
196     }
197
203     virtual void setCallBack(FilterCallback callback) {
204         callback_ = callback;
205         ob_error *error = nullptr;
206         ob_filter_set_callback(impl_, &Filter::filterCallback, this, &error);
207         Error::handle(&error);
208     }
209
217     virtual std::string getConfigSchema() const {
218         ob_error *error = nullptr;
219         auto schema = ob_filter_get_config_schema(impl_, &error);
220         Error::handle(&error);
221         return schema;
222     }
223
230     virtual std::vector<OBFilterConfigSchemaItem> getConfigSchemaVec() const {
231         return configSchemaVec_;
232     }
233
243     virtual void setConfigValue(const std::string &configName, double value) const {
244         ob_error *error = nullptr;
245         ob_filter_set_config_value(impl_, configName.c_str(), value, &error);
246         Error::handle(&error);
247     }
248
258     virtual double getConfigValue(const std::string &configName) const {
259         ob_error *error = nullptr;
260         double value = ob_filter_get_config_value(impl_, configName.c_str(), &error);
261         Error::handle(&error);
262         return value;
263     }
264
265 private:
266     static void filterCallback(ob_frame *frame, void *userData) {
267         auto filter = static_cast<Filter *>(userData);
268         filter->callback_(std::make_shared<Frame>(frame));
269     }
270
271 public:
272     // The following interfaces are deprecated and are retained here for compatibility purposes.
273     virtual const char *type() {
274         return getName().c_str();
275     }

```

```

279
280 template <typename T> bool is();
281
282 template <typename T> std::shared_ptr<T> as() {
283     if(!is<T>()) {
284         throw std::runtime_error("unsupported operation, object's type is not require type");
285     }
286
287     return std::static_pointer_cast<T>(shared_from_this());
288 }
289
290 class FilterFactory {
291 public:
292     static std::shared_ptr<Filter> createFilter(const std::string &name) {
293         ob_error *error=nullptr;
294         auto    impl = ob_create_filter(name.c_str(), &error);
295         Error::handle(&error);
296         return std::make_shared<Filter>(impl);
297     }
298
299     static std::shared_ptr<Filter> createPrivateFilter(const std::string &name, const std::string &activationK
300         ey) {
301         ob_error *error=nullptr;
302         auto    impl = ob_create_private_filter(name.c_str(), activationKey.c_str(), &error);
303         Error::handle(&error);
304         return std::make_shared<Filter>(impl);
305     }
306
307     static std::string getFilterVendorSpecificCode(const std::string &name) {
308         ob_error *error=nullptr;
309         auto    code = ob_filter_get_vendor_specific_code(name.c_str(), &error);
310         Error::handle(&error);
311         return code;
312     }
313 };
314
315 class PointCloudFilter : public Filter {
316 public:
317     PointCloudFilter() {
318         ob_error *error=nullptr;
319         auto    impl = ob_create_filter("PointCloudFilter", &error);
320         Error::handle(&error);
321         init(impl);
322     }
323
324     virtual ~PointCloudFilter() noexcept override = default;
325
326     void setCreatePointFormat(OBFormat format) {
327         setConfigValue("pointFormat", static_cast<double>(format));
328     }
329
330     void setCoordinateDataScaled(float factor) {
331         setConfigValue("coordinateDataScale", factor);
332     }
333
334     void setColorDataNormalization(bool state) {
335         setConfigValue("colorDataNormalization", state);
336     }
337
338     void setCoordinateSystem(OBCoordinateSystemType type) {
339         setConfigValue("coordinateSystemType", static_cast<double>(type));
340     }
341
342 public:
343     // The following interfaces are deprecated and are retained here for compatibility purposes.

```

```

396     void setPositionDataScaled(float scale) {
397         setCoordinateDataScaled(scale);
398     }
399
400     // The following interfaces are deprecated and are retained here for compatibility purposes.
401     void setFrameAlignState(bool state) {
402         (void)state; // to compile
403     }
404     // The following interfaces are deprecated and are retained here for compatibility purposes.
405     void setCameraParam(OBCameraParam param) {
406         (void)param;
407     }
408 };
409
413 class Align : public Filter {
414 public:
415     Align(OBStreamType alignToStreamType) {
416         ob_error *error=nullptr;
417         auto    impl = ob_create_filter("Align", &error);
418         Error::handle(&error);
419         init(impl);
420
421         setConfigValue("AlignType", static_cast<double>(alignToStreamType));
422     }
423
424     virtual ~Align() noexcept override = default;
425
426     OBStreamType getAlignToStreamType() {
427         return static_cast<OBStreamType>(static_cast<int>(getConfigValue("AlignType")));
428     }
429
440     void setMatchTargetResolution(bool state) {
441         setConfigValue("MatchTargetRes", state);
442     }
443
450     void setAlignToStreamProfile(std::shared_ptr<const StreamProfile> profile) {
451         ob_error *error=nullptr;
452         ob_align_filter_set_align_to_stream_profile(impl_, profile->getImpl(), &error);
453         Error::handle(&error);
454     }
455 };
456
460 class FormatConvertFilter : public Filter {
461 public:
462     FormatConvertFilter() {
463         ob_error *error=nullptr;
464         auto    impl = ob_create_filter("FormatConverter", &error);
465         Error::handle(&error);
466         init(impl);
467     }
468
469     virtual ~FormatConvertFilter() noexcept override = default;
470
476     void setFormatConvertType(OBConvertFormat type) {
477         setConfigValue("convertType", static_cast<double>(type));
478     }
479 };
480
486 class HdrMerge : public Filter {
487 public:
488     HdrMerge() {
489         ob_error *error=nullptr;
490         auto    impl = ob_create_filter("HDRMerge", &error);
491         Error::handle(&error);
492         init(impl);

```

```

493     }
494
495     virtual ~HdrMerge() noexcept override = default;
496 };
497
501 class SequenceIdFilter : public Filter {
502 private:
503     std::map<float, std::string> sequenceIdList_{ { 0.f, "all" }, { 1.f, "1" } };
504     OBSequenceIdItem      *outputSequenceIdList_ = nullptr;
505
506     void initSequenceIdList() {
507         outputSequenceIdList_ = new OBSequenceIdItem[sequenceIdList_.size()];
508
509         int i = 0;
510         for(const auto &pair: sequenceIdList_) {
511             outputSequenceIdList_[i].sequenceSelectId = static_cast<int>(pair.first);
512             strncpy(outputSequenceIdList_[i].name, pair.second.c_str(), sizeof(outputSequenceIdList_[i].name)
513 - 1);
513             outputSequenceIdList_[i].name[sizeof(outputSequenceIdList_[i].name) - 1] = '\0';
514             ++i;
515         }
516     }
517
518 public:
519     SequenceIdFilter() {
520         ob_error *error = nullptr;
521         auto    impl = ob_create_filter("SequenceIdFilter", &error);
522         Error::handle(&error);
523         init(impl);
524         initSequenceIdList();
525     }
526
527     virtual ~SequenceIdFilter() noexcept override {
528         if(outputSequenceIdList_) {
529             delete[] outputSequenceIdList_;
530             outputSequenceIdList_ = nullptr;
531         }
532     }
533
539     void selectSequenceId(int sequence_id) {
540         setConfigValue("sequenceid", static_cast<double>(sequence_id));
541     }
542
548     int getSelectSequenceId() {
549         return static_cast<int>(getConfigValue("sequenceid"));
550     }
551
552     OBSequenceIdItem *getSequenceIdList() {
553         return outputSequenceIdList_;
554     }
555
561     int getSequenceIdListSize() {
562         return static_cast<int>(sequenceIdList_.size());
563     }
564 };
565
569 class DecimationFilter : public Filter {
570 public:
571     DecimationFilter() {
572         ob_error *error = nullptr;
573         auto    impl = ob_create_filter("DecimationFilter", &error);
574         Error::handle(&error);
575         init(impl);
576     }
577
578     ...
579     ...
580     ...
581     ...
582     ...
583     ...
584     ...
585     ...
586     ...
587     ...
588     ...
589     ...
590     ...
591     ...
592     ...
593     ...
594     ...
595     ...
596     ...
597     ...
598     ...
599     ...
600     ...
601     ...
602     ...
603     ...
604     ...
605     ...
606     ...
607     ...
608     ...
609     ...
610     ...
611     ...
612     ...
613     ...
614     ...
615     ...
616     ...
617     ...
618     ...
619     ...
620     ...
621     ...
622     ...
623     ...
624     ...
625     ...
626     ...
627     ...
628     ...
629     ...
630     ...
631     ...
632     ...
633     ...
634     ...
635     ...
636     ...
637     ...
638     ...
639     ...
640     ...
641     ...
642     ...
643     ...
644     ...
645     ...
646     ...
647     ...
648     ...
649     ...
650     ...
651     ...
652     ...
653     ...
654     ...
655     ...
656     ...
657     ...
658     ...
659     ...
660     ...
661     ...
662     ...
663     ...
664     ...
665     ...
666     ...
667     ...
668     ...
669     ...
670     ...
671     ...
672     ...
673     ...
674     ...
675     ...
676     ...
677     ...
678     ...
679     ...
680     ...
681     ...
682     ...
683     ...
684     ...
685     ...
686     ...
687     ...
688     ...
689     ...
690     ...
691     ...
692     ...
693     ...
694     ...
695     ...
696     ...
697     ...
698     ...
699     ...
700     ...
701     ...
702     ...
703     ...
704     ...
705     ...
706     ...
707     ...
708     ...
709     ...
710     ...
711     ...
712     ...
713     ...
714     ...
715     ...
716     ...
717     ...
718     ...
719     ...
720     ...
721     ...
722     ...
723     ...
724     ...
725     ...
726     ...
727     ...
728     ...
729     ...
730     ...
731     ...
732     ...
733     ...
734     ...
735     ...
736     ...
737     ...
738     ...
739     ...
740     ...
741     ...
742     ...
743     ...
744     ...
745     ...
746     ...
747     ...
748     ...
749     ...
750     ...
751     ...
752     ...
753     ...
754     ...
755     ...
756     ...
757     ...
758     ...
759     ...
760     ...
761     ...
762     ...
763     ...
764     ...
765     ...
766     ...
767     ...
768     ...
769     ...
770     ...
771     ...
772     ...
773     ...
774     ...
775     ...
776     ...
777     ...
778     ...
779     ...
780     ...
781     ...
782     ...
783     ...
784     ...
785     ...
786     ...
787     ...
788     ...
789     ...
790     ...
791     ...
792     ...
793     ...
794     ...
795     ...
796     ...
797     ...
798     ...
799     ...
800     ...
801     ...
802     ...
803     ...
804     ...
805     ...
806     ...
807     ...
808     ...
809     ...
810     ...
811     ...
812     ...
813     ...
814     ...
815     ...
816     ...
817     ...
818     ...
819     ...
820     ...
821     ...
822     ...
823     ...
824     ...
825     ...
826     ...
827     ...
828     ...
829     ...
830     ...
831     ...
832     ...
833     ...
834     ...
835     ...
836     ...
837     ...
838     ...
839     ...
840     ...
841     ...
842     ...
843     ...
844     ...
845     ...
846     ...
847     ...
848     ...
849     ...
850     ...
851     ...
852     ...
853     ...
854     ...
855     ...
856     ...
857     ...
858     ...
859     ...
860     ...
861     ...
862     ...
863     ...
864     ...
865     ...
866     ...
867     ...
868     ...
869     ...
870     ...
871     ...
872     ...
873     ...
874     ...
875     ...
876     ...
877     ...
878     ...
879     ...
880     ...
881     ...
882     ...
883     ...
884     ...
885     ...
886     ...
887     ...
888     ...
889     ...
890     ...
891     ...
892     ...
893     ...
894     ...
895     ...
896     ...
897     ...
898     ...
899     ...
900     ...
901     ...
902     ...
903     ...
904     ...
905     ...
906     ...
907     ...
908     ...
909     ...
910     ...
911     ...
912     ...
913     ...
914     ...
915     ...
916     ...
917     ...
918     ...
919     ...
920     ...
921     ...
922     ...
923     ...
924     ...
925     ...
926     ...
927     ...
928     ...
929     ...
930     ...
931     ...
932     ...
933     ...
934     ...
935     ...
936     ...
937     ...
938     ...
939     ...
940     ...
941     ...
942     ...
943     ...
944     ...
945     ...
946     ...
947     ...
948     ...
949     ...
950     ...
951     ...
952     ...
953     ...
954     ...
955     ...
956     ...
957     ...
958     ...
959     ...
960     ...
961     ...
962     ...
963     ...
964     ...
965     ...
966     ...
967     ...
968     ...
969     ...
970     ...
971     ...
972     ...
973     ...
974     ...
975     ...
976     ...
977     ...
978     ...
979     ...
980     ...
981     ...
982     ...
983     ...
984     ...
985     ...
986     ...
987     ...
988     ...
989     ...
990     ...
991     ...
992     ...
993     ...
994     ...
995     ...
996     ...
997     ...
998     ...
999     ...

```

```

578     virtual ~DecimationFilter() noexcept override = default;
579
585     void setScaleValue(uint8_t value) {
586         setConfigValue("decimate", static_cast<double>(value));
587     }
588
592     uint8_t getScaleValue() {
593         return static_cast<uint8_t>(getConfigValue("decimate"));
594     }
595
599     OBUInt8PropertyRange getScaleRange() {
600         OBUInt8PropertyRange scaleRange{};
601         if(configSchemaVec_.size() != 0) {
602             const auto &item = configSchemaVec_[0];
603             scaleRange = getPropertyRange<OBUInt8PropertyRange>(item, getConfigValue("decimate"));
604         }
605         return scaleRange;
606     }
607 };
608
613 class ThresholdFilter : public Filter {
614 public:
615     ThresholdFilter() {
616         ob_error *error = nullptr;
617         auto impl = ob_create_filter("ThresholdFilter", &error);
618         Error::handle(&error);
619         init(impl);
620     }
621
622     virtual ~ThresholdFilter() noexcept override = default;
623
629     OBIntPropertyRange getMinRange() {
630         OBIntPropertyRange range{};
631         const auto &schemaVec = getConfigSchemaVec();
632         for(const auto &item: schemaVec) {
633             if(strcmp(item.name, "min") == 0) {
634                 range = getPropertyRange<OBIntPropertyRange>(item, getConfigValue("min"));
635                 break;
636             }
637         }
638         return range;
639     }
640
646     OBIntPropertyRange getMaxRange() {
647         OBIntPropertyRange range{};
648         const auto &schemaVec = getConfigSchemaVec();
649         for(const auto &item: schemaVec) {
650             if(strcmp(item.name, "max") == 0) {
651                 range = getPropertyRange<OBIntPropertyRange>(item, getConfigValue("max"));
652                 break;
653             }
654         }
655         return range;
656     }
657
661     bool setValueRange(uint16_t min, uint16_t max) {
662         if(min >= max) {
663             return false;
664         }
665         setConfigValue("min", min);
666         setConfigValue("max", max);
667         return true;
668     }
669 };
670
676 class SpatialAveragedFilter : public Filter {

```

```

5/5  class SpatialAdvancedFilter : public Filter {
677 public:
678     SpatialAdvancedFilter(const std::string &activationKey = "") {
679         ob_error *error = nullptr;
680         auto    impl = ob_create_private_filter("SpatialAdvancedFilter", activationKey.c_str(), &error);
681         Error::handle(&error);
682         init(impl);
683     }
684
685     virtual ~SpatialAdvancedFilter() noexcept override = default;
686
692     OBFloatPropertyRange getAlphaRange() {
693         OBFloatPropertyRange range{};
694         const auto    &schemaVec = getConfigSchemaVec();
695         for(const auto &item: schemaVec) {
696             if(strcmp(item.name, "alpha") == 0) {
697                 range = getPropertyRange<OBFloatPropertyRange>(item, getConfigValue("alpha"));
698                 break;
699             }
700         }
701         return range;
702     }
703
709     OBUInt16PropertyRange getDispDiffRange() {
710         OBUInt16PropertyRange range{};
711         const auto    &schemaVec = getConfigSchemaVec();
712         for(const auto &item: schemaVec) {
713             if(strcmp(item.name, "disp_diff") == 0) {
714                 range = getPropertyRange<OBUInt16PropertyRange>(item, getConfigValue("disp_diff"));
715                 break;
716             }
717         }
718         return range;
719     }
720
726     OBUInt16PropertyRange getRadiusRange() {
727         OBUInt16PropertyRange range{};
728         const auto    &schemaVec = getConfigSchemaVec();
729         for(const auto &item: schemaVec) {
730             if(strcmp(item.name, "radius") == 0) {
731                 range = getPropertyRange<OBUInt16PropertyRange>(item, getConfigValue("radius"));
732                 break;
733             }
734         }
735         return range;
736     }
737
743     OBIntPropertyRange getMagnitudeRange() {
744         OBIntPropertyRange range{};
745         const auto    &schemaVec = getConfigSchemaVec();
746         for(const auto &item: schemaVec) {
747             if(strcmp(item.name, "magnitude") == 0) {
748                 range = getPropertyRange<OBIntPropertyRange>(item, getConfigValue("magnitude"));
749                 break;
750             }
751         }
752         return range;
753     }
754
760     OBSpatialAdvancedFilterParams getFilterParams() {
761         OBSpatialAdvancedFilterParams params{};
762         params.alpha   = static_cast<float>(getConfigValue("alpha"));
763         params.disp_diff = static_cast<uint16_t>(getConfigValue("disp_diff"));
764         params.magnitude = static_cast<uint8_t>(getConfigValue("magnitude"));
765         params.radius   = static_cast<uint16_t>(getConfigValue("radius"));
766         return params;

```

```

767     }
768
774 void setFilterParams(OBSpatialAdvancedFilterParams params) {
775     setConfigValue("alpha", params.alpha);
776     setConfigValue("disp_diff", params.disp_diff);
777     setConfigValue("magnitude", params.magnitude);
778     setConfigValue("radius", params.radius);
779 }
780 };
781
785 class HoleFillingFilter : public Filter {
786 public:
787     HoleFillingFilter(const std::string &activationKey = "") {
788         ob_error *error = nullptr;
789         auto impl = ob_create_private_filter("HoleFillingFilter", activationKey.c_str(), &error);
790         Error::handle(&error);
791         init(impl);
792     }
793
794     ~HoleFillingFilter() noexcept override = default;
795
801     void setFilterMode(OBHoleFillingMode mode) {
802         setConfigValue("hole_filling_mode", static_cast<double>(mode));
803     }
804
810     OBHoleFillingMode getFilterMode() {
811         return static_cast<OBHoleFillingMode>(static_cast<int>(getConfigValue("hole_filling_mode")));
812     }
813 };
814
818 class NoiseRemovalFilter : public Filter {
819 public:
820     NoiseRemovalFilter(const std::string &activationKey = "") {
821         ob_error *error = nullptr;
822         auto impl = ob_create_private_filter("NoiseRemovalFilter", activationKey.c_str(), &error);
823         Error::handle(&error);
824         init(impl);
825     }
826
827     ~NoiseRemovalFilter() noexcept override = default;
828
834     void setFilterParams(OBNoiseRemovalFilterParams filterParams) {
835         setConfigValue("max_size", static_cast<double>(filterParams.max_size));
836         setConfigValue("min_diff", static_cast<double>(filterParams.disp_diff));
837         // todo:set noise remove type
838     }
839
845     OBNoiseRemovalFilterParams getFilterParams() {
846         OBNoiseRemovalFilterParams param{};
847         param.max_size = static_cast<uint16_t>(getConfigValue("max_size"));
848         param.disp_diff = static_cast<uint16_t>(getConfigValue("min_diff"));
849         // todo: type is not set
850         return param;
851     }
852
857     OBUint16PropertyRange getDispDiffRange() {
858         OBUint16PropertyRange range{};
859         const auto &schemaVec = getConfigSchemaVec();
860         for(const auto &item: schemaVec) {
861             if(strcmp(item.name, "min_diff") == 0) {
862                 range = getPropertyRange<OBUint16PropertyRange>(item, getConfigValue("min_diff"));
863                 break;
864             }
865         }
866         return range;

```

```

867 }
868
873 OB UInt16PropertyRange getMaxSizeRange() {
874     OB UInt16PropertyRange range {};
875     const auto &schemaVec = getConfigSchemaVec();
876     for(const auto &item: schemaVec) {
877         if(strcmp(item.name, "max_size") == 0) {
878             range = getPropertyRange<OB UInt16PropertyRange>(item, getConfigValue("max_size"));
879             break;
880         }
881     }
882     return range;
883 }
884 };
885
889 class TemporalFilter : public Filter {
890 public:
891     TemporalFilter(const std::string &activationKey = "") {
892         ob_error *error = nullptr;
893         auto impl = ob_create_private_filter("TemporalFilter", activationKey.c_str(), &error);
894         Error::handle(&error);
895         init(impl);
896     }
897
898     ~TemporalFilter() noexcept override = default;
899
905     OB FloatPropertyRange getDiffScaleRange() {
906         OB FloatPropertyRange range {};
907         const auto &schemaVec = getConfigSchemaVec();
908         for(const auto &item: schemaVec) {
909             if(strcmp(item.name, "diff_scale") == 0) {
910                 range = getPropertyRange<OB FloatPropertyRange>(item, getConfigValue("diff_scale"));
911                 break;
912             }
913         }
914         return range;
915     }
916
922     void setDiffScale(float value) {
923         setConfigValue("diff_scale", static_cast<double>(value));
924     }
925
931     OB FloatPropertyRange getWeightRange() {
932         OB FloatPropertyRange range {};
933         const auto &schemaVec = getConfigSchemaVec();
934         for(const auto &item: schemaVec) {
935             if(strcmp(item.name, "weight") == 0) {
936                 range = getPropertyRange<OB FloatPropertyRange>(item, getConfigValue("weight"));
937                 break;
938             }
939         }
940         return range;
941     }
942
948     void setWeight(float value) {
949         setConfigValue("weight", static_cast<double>(value));
950     }
951 };
952
956 class DisparityTransform : public Filter {
957 public:
958     DisparityTransform(const std::string &activationKey = "") {
959         ob_error *error = nullptr;
960         auto impl = ob_create_private_filter("DisparityTransform", activationKey.c_str(), &error);
961         Error::handle(&error);
962     }

```

```

962     init(impl);
963 }
964
965 ~DisparityTransform() noexcept override = default;
966 };
967
968 class OBFilterList {
969 private:
970     ob_filter_list_t *impl_;
971
972 public:
973     explicit OBFilterList(ob_filter_list_t *impl) : impl_(impl) {}
974
975     ~OBFilterList() noexcept {
976         ob_error *error = nullptr;
977         ob_delete_filter_list(impl_, &error);
978         Error::handle(&error, false);
979     }
980
981     uint32_t getCount() const {
982         ob_error *error = nullptr;
983         auto count = ob_filter_list_get_count(impl_, &error);
984         Error::handle(&error);
985         return count;
986     }
987
988     std::shared_ptr<Filter> getFilter(uint32_t index) {
989         ob_error *error = nullptr;
990         auto filter = ob_filter_list_get_filter(impl_, index, &error);
991         Error::handle(&error);
992         return std::make_shared<Filter>(filter);
993     }
994
995 public:
996     // The following interfaces are deprecated and are retained here for compatibility purposes.
997     uint32_t count() const {
998         return getCount();
999     }
1000
1001 static const std::unordered_map<std::string, std::type_index> obFilterTypeMap = {
1002     { "PointCloudFilter", typeid(PointCloudFilter) }, { "Align", typeid(Align) },
1003     { "FormatConverter", typeid(FormatConvertFilter) }, { "HDRMerge", typeid(HdrMerge) },
1004     { "SequenceIdFilter", typeid(SequenceIdFilter) }, { "DecimationFilter", typeid(DecimationFilter) },
1005     { "ThresholdFilter", typeid(ThresholdFilter) }, { "SpatialAdvancedFilter", typeid(SpatialAdvancedFilter) },
1006     { "HoleFillingFilter", typeid(HoleFillingFilter) }, { "NoiseRemovalFilter", typeid(NoiseRemovalFilter) },
1007     { "TemporalFilter", typeid(TemporalFilter) }, { "DisparityTransform", typeid(DisparityTransform) }
1008 };
1009
1010 template <typename T> bool Filter::is() {
1011     std::string name = type();
1012     auto it = obFilterTypeMap.find(name);
1013     if(it != obFilterTypeMap.end()) {
1014         return std::type_index(typeid(T)) == it->second;
1015     }
1016     return false;
1017 }
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027 } // namespace ob

```

# Frame.h File Reference

Frame related function is mainly used to obtain frame data and frame information. [More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_video_frame_metadata ob_frame_get_metadata
#define ob_video_frame_metadata_size ob_frame_get_metadata_size
#define ob_frame_index ob_frame_get_index
#define ob_frame_format ob_frame_get_format
#define ob_frame_time_stamp_us ob_frame_get_timestamp_us
#define ob_frame_set_device_time_stamp_us ob_frame_set_timestamp_us
#define ob_frame_system_time_stamp_us ob_frame_get_system_timestamp_us
#define ob_frame_global_time_stamp_us ob_frame_get_global_timestamp_us
#define ob_frame_data ob_frame_get_data
#define ob_frame_data_size ob_frame_get_data_size
#define ob_frame_metadata ob_frame_get_metadata
#define ob_frame_metadata_size ob_frame_get_metadata_size
#define ob_video_frame_width ob_video_frame_get_width
#define ob_video_frame_height ob_video_frame_get_height
#define ob_video_frame_pixel_available_bit_size ob_video_frame_get_pixel_available_bit_size
#define ob_points_frame_get_position_value_scale ob_points_frame_get_coordinate_value_scale
#define ob_frameset_frame_count ob_frameset_get_count
#define ob_frameset_depth_frame ob_frameset_get_depth_frame
#define ob_frameset_color_frame ob_frameset_get_color_frame
#define ob_frameset_ir_frame ob_frameset_get_ir_frame
#define ob_frameset_points_frame ob_frameset_get_points_frame
#define ob_accel_frame_value ob_accel_frame_get_value
#define ob_accel_frame_temperature ob_accel_frame_get_temperature
#define ob_gyro_frame_value ob_gyro_frame_get_value
#define ob_gyro_frame_temperature ob_gyro_frame_get_temperature
#define ob_frameset_get_frame_count ob_frameset_get_count
#define ob_frame_time_stamp(frame, err)
#define ob_frame_system_time_stamp(frame, err)
```

```
#define ob_frame_set_system_time_stamp(frame, system_timestamp, err)
#define ob_frame_set_device_time_stamp(frame, device_timestamp, err)
```

## Functions

```
OB_EXPORT ob_frame * ob_create_frame (ob_frame_type frame_type, ob_format format,
                                         uint32_t data_size, ob_error **error)
                                         Create a frame object based on the specified parameters.
```

```
OB_EXPORT ob_frame * ob_create_frame_from_other_frame (const ob_frame
                                         *other_frame, bool should_copy_data, ob_error **error)
                                         Create (clone) a frame object based on the specified other frame
                                         object.
```

```
OB_EXPORT ob_frame * ob_create_frame_from_stream_profile (const
                                         ob_stream_profile *stream_profile, ob_error **error)
                                         Create a frame object according to the specified stream profile.
```

```
OB_EXPORT ob_frame * ob_create_video_frame (ob_frame_type frame_type, ob_format
                                         format, uint32_t width, uint32_t height, uint32_t stride_bytes,
                                         ob_error **error)
                                         Create an video frame object based on the specified parameters.
```

```
OB_EXPORT ob_frame * ob_create_frame_from_buffer (ob_frame_type frame_type,
                                         ob_format format, uint8_t *buffer, uint32_t buffer_size,
                                         ob_frame_destroy_callback *buffer_destroy_cb, void
                                         *buffer_destroy_context, ob_error **error)
                                         Create a frame object based on an externally created buffer.
```

```
OB_EXPORT ob_frame * ob_create_video_frame_from_buffer (ob_frame_type
                                         frame_type, ob_format format, uint32_t width, uint32_t height,
                                         uint32_t stride_bytes, uint8_t *buffer, uint32_t buffer_size,
                                         ob_frame_destroy_callback *buffer_destroy_cb, void
                                         *buffer_destroy_context, ob_error **error)
                                         Create a video frame object based on an externally created buffer.
```

```
OB_EXPORT ob_frame * ob_create_frameset (ob_error **error)
                                         Create an empty frameset object.
```

```
OB_EXPORT void ob_frame_add_ref (const ob_frame *frame, ob_error **error)
                                         Increase the reference count of a frame object.
```

```
OB_EXPORT void ob_delete_frame (const ob_frame *frame, ob_error **error)
                                         Delete a frame object.
```

```
OB_EXPORT void ob_frame_copy_info (const ob_frame *src_frame, ob_frame
                                         *dst_frame, ob_error **error)
                                         Copy the information of the source frame object to the destination
                                         frame object.
```

**OB\_EXPORT** uint64\_t **ob\_frame\_get\_index** (const **ob\_frame** \*frame, **ob\_error** \*\*error)  
Get the frame index.

**OB\_EXPORT** **ob\_format** **ob\_frame\_get\_format** (const **ob\_frame** \*frame, **ob\_error** \*\*error)  
Get the frame format.

**OB\_EXPORT** **ob\_frame\_type** **ob\_frame\_get\_type** (const **ob\_frame** \*frame, **ob\_error** \*\*error)  
Get the frame type.

**OB\_EXPORT** uint64\_t **ob\_frame\_get\_timestamp\_us** (const **ob\_frame** \*frame, **ob\_error** \*\*error)  
Get the frame timestamp (also known as device timestamp, hardware timestamp) of the frame in microseconds.

**OB\_EXPORT** void **ob\_frame\_set\_timestamp\_us** (**ob\_frame** \*frame, uint64\_t timestamp\_us, **ob\_error** \*\*error)

Set the frame timestamp (also known as the device timestamp, hardware timestamp) of a frame object.

**OB\_EXPORT** uint64\_t **ob\_frame\_get\_system\_timestamp\_us** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the system timestamp of the frame in microseconds.

**OB\_EXPORT** void **ob\_frame\_set\_system\_timestamp\_us** (**ob\_frame** \*frame, uint64\_t system\_timestamp\_us, **ob\_error** \*\*error)

Set the system timestamp of the frame in microseconds.

**OB\_EXPORT** uint64\_t **ob\_frame\_get\_global\_timestamp\_us** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the global timestamp of the frame in microseconds.

**OB\_EXPORT** uint8\_t \* **ob\_frame\_get\_data** (const **ob\_frame** \*frame, **ob\_error** \*\*error)  
Get the data buffer of a frame.

**OB\_EXPORT** void **ob\_frame\_update\_data** (**ob\_frame** \*frame, const uint8\_t \*data, uint32\_t data\_size, **ob\_error** \*\*error)  
Update the data of a frame.

**OB\_EXPORT** uint32\_t **ob\_frame\_get\_data\_size** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the frame data size.

**OB\_EXPORT** uint8\_t \* **ob\_frame\_get\_metadata** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the metadata of the frame.

**OB\_EXPORT** uint32\_t **ob\_frame\_get\_metadata\_size** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the metadata size of the frame.

**OB\_EXPORT** void **ob\_frame\_update\_metadata** (**ob\_frame** \*frame, const uint8\_t \*metadata, uint32\_t metadata\_size, **ob\_error** \*\*error)

Update the metadata of the frame.

**OB\_EXPORT** bool **ob\_frame\_has\_metadata** (const **ob\_frame** \*frame,  
**ob\_frame\_metadata\_type** type, **ob\_error** \*\*error)

check if the frame contains the specified metadata

**OB\_EXPORT** int64\_t **ob\_frame\_get\_metadata\_value** (const **ob\_frame** \*frame,  
**ob\_frame\_metadata\_type** type, **ob\_error** \*\*error)

Get the metadata value of the frame.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_frame\_get\_stream\_profile** (const **ob\_frame** \*frame, **ob\_error**  
\*\*error)

Get the stream profile of the frame.

**OB\_EXPORT** void **ob\_frame\_set\_stream\_profile** (**ob\_frame** \*frame, const  
**ob\_stream\_profile** \*stream\_profile, **ob\_error** \*\*error)

Set (override) the stream profile of the frame.

**OB\_EXPORT** **ob\_sensor** \* **ob\_frame\_get\_sensor** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the sensor of the frame.

**OB\_EXPORT** **ob\_device** \* **ob\_frame\_get\_device** (const **ob\_frame** \*frame, **ob\_error** \*\*error)

Get the device of the frame.

**OB\_EXPORT** uint32\_t **ob\_video\_frame\_get\_width** (const **ob\_frame** \*frame, **ob\_error**  
\*\*error)

Get video frame width.

**OB\_EXPORT** uint32\_t **ob\_video\_frame\_get\_height** (const **ob\_frame** \*frame, **ob\_error**  
\*\*error)

Get video frame height.

**OB\_EXPORT** **ob\_pixel\_type** **ob\_video\_frame\_get\_pixel\_type** (const **ob\_frame** \*frame,  
**ob\_error** \*\*error)

Get video frame pixel format.

**OB\_EXPORT** void **ob\_video\_frame\_set\_pixel\_type** (**ob\_frame** \*frame,  
**ob\_pixel\_type** pixel\_type, **ob\_error** \*\*error)

Set video frame pixel format.

**OB\_EXPORT** uint8\_t **ob\_video\_frame\_get\_pixel\_available\_bit\_size** (const **ob\_frame**  
\*frame, **ob\_error** \*\*error)

Get the effective number of pixels (such as Y16 format frame, but  
only the lower 10 bits are effective bits, and the upper 6 bits are  
filled with 0)

**OB\_EXPORT** void **ob\_video\_frame\_set\_pixel\_available\_bit\_size** (**ob\_frame** \*frame,  
uint8\_t bit\_size, **ob\_error** \*\*error)

Set the effective number of pixels (such as Y16 format frame, but  
only the lower 10 bits are effective bits, and the upper 6 bits are  
filled with 0)

**OB\_EXPORT** **ob\_sensor\_type** **ob\_ir\_frame\_get\_source\_sensor\_type** (const **ob\_frame** \*frame,  
**ob\_error** \*\***ob\_error**)

Get the source sensor type of the ir frame (left or right for dual camera)

**OB\_EXPORT** float **ob\_depth\_frame\_get\_value\_scale** (const **ob\_frame** \*frame,  
**ob\_error** \*\*error)

Get the value scale of the depth frame. The pixel value of the depth frame is multiplied by the scale to give a depth value in millimeters. For example, if valueScale=0.1 and a certain coordinate pixel value is pixelValue=10000, then the depth value = pixelValue\*valueScale = 10000\*0.1=1000mm.

**OB\_EXPORT** void **ob\_depth\_frame\_set\_value\_scale** (**ob\_frame** \*frame, float  
value\_scale, **ob\_error** \*\*error)

Set the value scale of the depth frame. The pixel value of the depth frame is multiplied by the scale to give a depth value in millimeters. For example, if valueScale=0.1 and a certain coordinate pixel value is pixelValue=10000, then the depth value = pixelValue\*valueScale = 10000\*0.1=1000mm.

**OB\_EXPORT** float **ob\_points\_frame\_get\_coordinate\_value\_scale** (const **ob\_frame**  
\*frame, **ob\_error** \*\*error)

Get the point coordinate value scale of the points frame. The point position value of the points frame is multiplied by the scale to give a position value in millimeters. For example, if scale=0.1, the x-coordinate value of a point is x = 10000, which means that the actual x-coordinate value = x\*scale = 10000\*0.1 = 1000mm.

**OB\_EXPORT** **ob\_accel\_value** **ob\_accel\_frame\_get\_value** (const **ob\_frame** \*frame, **ob\_error**  
\*\*error)

Get accelerometer frame data.

**OB\_EXPORT** float **ob\_accel\_frame\_get\_temperature** (const **ob\_frame** \*frame,  
**ob\_error** \*\*error)

Get the temperature when acquiring the accelerometer frame.

**OB\_EXPORT** **ob\_gyro\_value** **ob\_gyro\_frame\_get\_value** (const **ob\_frame** \*frame, **ob\_error**  
\*\*error)

Get gyroscope frame data.

**OB\_EXPORT** float **ob\_gyro\_frame\_get\_temperature** (const **ob\_frame** \*frame,  
**ob\_error** \*\*error)

Get the temperature when acquiring the gyroscope frame.

**OB\_EXPORT** uint32\_t **ob\_frameset\_get\_count** (const **ob\_frame** \*frameset, **ob\_error**  
\*\*error)

Get the number of frames contained in the frameset.

```

OB_EXPORT ob_frame * ob_frameset_get_depth_frame (const ob_frame *frameset,
ob_error **error)
    Get the depth frame from the frameset.

OB_EXPORT ob_frame * ob_frameset_get_color_frame (const ob_frame *frameset,
ob_error **error)
    Get the color frame from the frameset.

OB_EXPORT ob_frame * ob_frameset_get_ir_frame (const ob_frame *frameset, ob_error
**error)
    Get the infrared frame from the frameset.

OB_EXPORT ob_frame * ob_frameset_get_points_frame (const ob_frame *frameset,
ob_error **error)
    Get point cloud frame from the frameset.

OB_EXPORT ob_frame * ob_frameset_get_frame (const ob_frame *frameset,
ob_frame_type frame_type, ob_error **error)
    Get a frame of a specific type from the frameset.

OB_EXPORT ob_frame * ob_frameset_get_frame_by_index (const ob_frame *frameset,
uint32_t index, ob_error **error)
    Get a frame at a specific index from the FrameSet.

OB_EXPORT void ob_frameset_push_frame (ob_frame *frameset, const ob_frame
*frame, ob_error **error)
    Push a frame to the frameset.

OB_EXPORT uint32_t ob_point_cloud_frame_get_width (const ob_frame *frame,
ob_error **error)
    Get point cloud frame width.

OB_EXPORT uint32_t ob_point_cloud_frame_get_height (const ob_frame *frame,
ob_error **error)
    Get point cloud frame height.

```

## Detailed Description

Frame related function is mainly used to obtain frame data and frame information.

Definition in file **Frame.h**.

## Macro Definition Documentation

- ◆ **ob\_video\_frame\_metadata**

```
#define ob_video_frame_metadata ob_frame_get_metadata
```

Definition at line **292** of file **Frame.h**.

◆ **ob\_video\_frame\_metadata\_size**

```
#define ob_video_frame_metadata_size ob_frame_get_metadata_size
```

Definition at line **302** of file **Frame.h**.

◆ **ob\_frame\_index**

```
#define ob_frame_index ob_frame_get_index
```

Definition at line **625** of file **Frame.h**.

◆ **ob\_frame\_format**

```
#define ob_frame_format ob_frame_get_format
```

Definition at line **626** of file **Frame.h**.

◆ **ob\_frame\_time\_stamp\_us**

```
#define ob_frame_time_stamp_us ob_frame_get_timestamp_us
```

Definition at line **627** of file **Frame.h**.

◆ **ob\_frame\_set\_device\_time\_stamp\_us**

```
#define ob_frame_set_device_time_stamp_us ob_frame_set_timestamp_us
```

Definition at line **628** of file **Frame.h**.

◆ **ob\_frame\_system\_time\_stamp\_us**

```
#define ob_frame_system_time_stamp_us ob_frame_get_system_timestamp_us
```

Definition at line **629** of file **Frame.h**.

◆ **ob\_frame\_global\_time\_stamp\_us**

```
#define ob_frame_global_time_stamp_us ob_frame_get_global_timestamp_us
```

Definition at line **630** of file **Frame.h**.

◆ **ob\_frame\_data**

```
#define ob_frame_data ob_frame_get_data
```

Definition at line **631** of file **Frame.h**.

◆ **ob\_frame\_data\_size**

```
#define ob_frame_data_size ob_frame_get_data_size
```

Definition at line **632** of file **Frame.h**.

◆ **ob\_frame\_metadata**

```
#define ob_frame_metadata ob_frame_get_metadata
```

Definition at line **633** of file **Frame.h**.

◆ **ob\_frame\_metadata\_size**

```
#define ob_frame_metadata_size ob_frame_get_metadata_size
```

Definition at line **634** of file **Frame.h**.

◆ **ob\_video\_frame\_width**

```
#define ob_video_frame_width ob_video_frame_get_width
```

Definition at line **635** of file **Frame.h**.

◆ **ob\_video\_frame\_height**

```
#define ob_video_frame_height ob_video_frame_get_height
```

Definition at line **636** of file **Frame.h**.

◆ **ob\_video\_frame\_pixel\_available\_bit\_size**

```
#define ob_video_frame_pixel_available_bit_size ob_video_frame_get_pixel_available_bit_size
```

Definition at line **637** of file **Frame.h**.

◆ **ob\_points\_frame\_get\_position\_value\_scale**

```
#define ob_points_frame_get_position_value_scale ob_points_frame_get_coordinate_value_scale
```

Definition at line **638** of file **Frame.h**.

◆ **ob\_frameset\_frame\_count**

```
#define ob_frameset_frame_count ob_frameset_get_count
```

Definition at line **639** of file **Frame.h**.

◆ **ob\_frameset\_depth\_frame**

```
#define ob_frameset_depth_frame ob_frameset_get_depth_frame
```

Definition at line **640** of file **Frame.h**.

◆ **ob\_frameset\_color\_frame**

```
#define ob_frameset_color_frame ob_frameset_get_color_frame
```

Definition at line **641** of file **Frame.h**.

◆ **ob\_frameset\_ir\_frame**

```
#define ob_frameset_ir_frame ob_frameset_get_ir_frame
```

Definition at line **642** of file **Frame.h**.

◆ **ob\_frameset\_points\_frame**

```
#define ob_frameset_points_frame ob_frameset_get_points_frame
```

Definition at line **643** of file **Frame.h**.

◆ **ob\_accel\_frame\_value**

```
#define ob_accel_frame_value ob_accel_frame_get_value
```

Definition at line **644** of file **Frame.h**.

◆ **ob\_accel\_frame\_temperature**

```
#define ob_accel_frame_temperature ob_accel_frame_get_temperature
```

Definition at line **645** of file **Frame.h**.

◆ **ob\_gyro\_frame\_value**

```
#define ob_gyro_frame_value ob_gyro_frame_get_value
```

Definition at line **646** of file **Frame.h**.

◆ **ob\_gyro\_frame\_temperature**

```
#define ob_gyro_frame_temperature ob\_gyro\_frame\_get\_temperature
```

Definition at line [647](#) of file [Frame.h](#).

◆ [ob\\_frameset\\_get\\_frame\\_count](#)

```
#define ob_frameset_get_frame_count ob\_frameset\_get\_count
```

Definition at line [648](#) of file [Frame.h](#).

◆ [ob\\_frame\\_time\\_stamp](#)

```
#define ob_frame_time_stamp ( frame,  
                           err )
```

**Value:**

```
(ob\_frame\_get\_timestamp\_us(frame, err) / 1000)
```

Definition at line [650](#) of file [Frame.h](#).

◆ [ob\\_frame\\_system\\_time\\_stamp](#)

```
#define ob_frame_system_time_stamp ( frame,  
                                    err )
```

**Value:**

```
(ob\_frame\_get\_system\_timestamp\_us(frame, err))
```

Definition at line [651](#) of file [Frame.h](#).

◆ [ob\\_frame\\_set\\_system\\_time\\_stamp](#)

```
#define ob_frame_set_system_time_stamp ( frame,  
                                         system_timestamp,  
                                         err )
```

**Value:**

([ob\\_frame\\_set\\_system\\_timestamp\\_us](#)(frame, system\_timestamp \* 1000, err))

Definition at line [652](#) of file [Frame.h](#).

◆ [ob\\_frame\\_set\\_device\\_time\\_stamp](#)

```
#define ob_frame_set_device_time_stamp ( frame,  
                                         device_timestamp,  
                                         err )
```

**Value:**

([ob\\_frame\\_set\\_timestamp\\_us](#)(frame, device\_timestamp \* 1000, err))

Definition at line [653](#) of file [Frame.h](#).

## Function Documentation

◆ [ob\\_create\\_frame\(\)](#)

```
OB_EXPORT ob_frame* ob_create_frame ( ob_frame_type frame_type,  
                                     ob_format      format,  
                                     uint32_t       data_size,  
                                     ob_error **   error )
```

Create a frame object based on the specified parameters.

### Attention

The frame object is created with a reference count of 1, and the reference count should be decreased by calling [ob\\_delete\\_frame\(\)](#) when it is no longer needed.

### Parameters

**frame\_type** The frame object type.  
**format** The frame object format.  
**data\_size** The size of the frame object data.  
**error** Pointer to an error object that will be set if an error occurs.

### Returns

ob\_frame\* Return the frame object.

Referenced by [ob::FrameFactory::createFrame\(\)](#).

- ◆ [ob\\_create\\_frame\\_from\\_other\\_frame\(\)](#)

Create (clone) a frame object based on the specified other frame object.

The new frame object will have the same properties as the other frame object, but the data buffer is newly allocated.

## Attention

The frame object is created with a reference count of 1, and the reference count should be decreased by calling `ob_delete_frame()` when it is no longer needed.

## Parameters

[in] <b>other_frame</b>	The frame object to create the new frame object according to.
[in] <b>should_copy_data</b>	If true, the data of the source frame object will be copied to the new frame object. If false, the new frame object will have a data buffer with random data.
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

## Returns

**ob frame\*** Return the new frame object.

Referenced by [ob::FrameFactory::createFrameFromOtherFrame\(\)](#).

- #### ◆ ob\_create\_frame\_from\_stream\_profile()

Create a frame object according to the specified stream profile.

## Attention

The frame object is created with a reference count of 1, and the reference count should be decreased by calling `ob_delete_frame()` when it is no longer needed.

## Parameters

**stream\_profile** The stream profile to create the new frame object according to.

**error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_frame\*** Return the new frame object.

Referenced by [ob::FrameFactory::createFrameFromStreamProfile\(\)](#).

- #### ◆ ob\_create\_video\_frame()

```
OB_EXPORT ob_frame* ob_create_video_frame ( ob_frame_type frame_type,  
                                              ob_format      format,  
                                              uint32_t        width,  
                                              uint32_t        height,  
                                              uint32_t        stride_bytes,  
                                              ob_error **   error )
```

Create an video frame object based on the specified parameters.

### Attention

The frame object is created with a reference count of 1, and the reference count should be decreased by calling **ob\_delete\_frame()** when it is no longer needed.

### Parameters

- [in] **frame\_type** Frame object type.
- [in] **format** Frame object format.
- [in] **width** Frame object width.
- [in] **height** Frame object height.
- [in] **stride\_bytes** Row span in bytes. If 0, the stride is calculated based on the width and format.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

ob\_frame\* Return an empty frame object.

Referenced by **ob::FrameFactory::createVideoFrame()**.

- ◆ **ob\_create\_frame\_from\_buffer()**

```
OB_EXPORT ob_frame * ob_create_frame_from_buffer ( ob_frame_type
                                                ob_format
                                                uint8_t *
                                                uint32_t
                                                ob_frame_destroy_callback * buffer_destroy_cb,
                                                void *
                                                ob_error **
                                                frame_type,
                                                format,
                                                buffer,
                                                buffer_size,
                                                buffer_destroy_cb,
                                                buffer_destroy_context,
                                                error )
```

Create a frame object based on an externally created buffer.

### Attention

The buffer is owned by the user and will not be destroyed by the frame object. The user should ensure that the buffer is valid and not modified.

The frame object is created with a reference count of 1, and the reference count should be decreased by calling [ob\\_delete\\_frame\(\)](#) when it is no longer needed.

### Parameters

[in] <b>frame_type</b>	Frame object type.
[in] <b>format</b>	Frame object format.
[in] <b>buffer</b>	Frame object buffer.
[in] <b>buffer_size</b>	Frame object buffer size.
[in] <b>buffer_destroy_cb</b>	Destroy callback, will be called when the frame object is destroyed.
[in] <b>buffer_destroy_context</b>	Destroy context, user-defined context to be passed to the destroy callback.
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

`ob_frame*` Return the frame object.

Referenced by [ob::FrameFactory::createFrameFromBuffer\(\)](#).

- ◆ [ob\\_create\\_video\\_frame\\_from\\_buffer\(\)](#)

```

OB_EXPORT ob_frame *
ob_create_video_frame_from_buffer
(
    ob_frame_type frame_type,
    ob_format format,
    uint32_t width,
    uint32_t height,
    uint32_t stride_bytes,
    uint8_t * buffer,
    uint32_t buffer_size,
    ob_frame_destroy_callback * buffer_destroy_cb,
    void * buffer_destroy_context,
    ob_error ** error
)

```

Create a video frame object based on an externally created buffer.

### Attention

The buffer is owned by the user and will not be destroyed by the frame object. The user should ensure that the buffer is valid and not modified.

The frame object is created with a reference count of 1, and the reference count should be decreased by calling [ob\\_delete\\_frame\(\)](#) when it is no longer needed.

### Parameters

[in] <b>frame_type</b>	Frame object type.
[in] <b>format</b>	Frame object format.
[in] <b>width</b>	Frame object width.
[in] <b>height</b>	Frame object height.
[in] <b>stride_bytes</b>	Row span in bytes. If 0, the stride is calculated based on the width and format.
[in] <b>buffer</b>	Frame object buffer.
[in] <b>buffer_size</b>	Frame object buffer size.
[in] <b>buffer_destroy_cb</b>	Destroy callback, user-defined function to destroy the buffer.
[in] <b>buffer_destroy_context</b>	Destroy context, user-defined context to be passed to the destroy callback.
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

ob\_frame\* Return the frame object.

Referenced by [ob::FrameFactory::createVideoFrameFromBuffer\(\)](#).

◆ **ob\_create\_frameset()**

**OB\_EXPORT** **ob\_frame** \* ob\_create\_frameset ( **ob\_error** \*\* error )

Create an empty frameset object.

A frameset object is a special type of frame object that can be used to store multiple frames.

**Attention**

The frameset object is created with a reference count of 1, and the reference count should be decreased by calling **ob\_delete\_frame()** when it is no longer needed.

**Parameters**

[out] **error** Pointer to an error object that will be set if an error occurs.

**Returns**

**ob\_frame**\* Return the frameset object.

◆ **ob\_frame\_add\_ref()**

**OB\_EXPORT** void ob\_frame\_add\_ref ( const **ob\_frame** \* frame,  
                                 **ob\_error** \*\*          error )

Increase the reference count of a frame object.

The reference count is used to manage the lifetime of the frame object.

**Attention**

When calling this function, the reference count of the frame object is increased and requires to be decreased by calling **ob\_delete\_frame()**.

**Parameters**

[in] **frame** Frame object to increase the reference count.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Frame::as()**, and **ob::Frame::as()**.

◆ **ob\_delete\_frame()**

```
OB_EXPORT void ob_delete_frame ( const ob_frame * frame,  
                                ob_error **      error )
```

Delete a frame object.

This function will decrease the reference count of the frame object and release the memory if the reference count becomes 0.

#### Parameters

- [in] **frame** The frame object to delete.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Frame::~Frame\(\)](#).

#### ◆ [ob\\_frame\\_copy\\_info\(\)](#)

```
OB_EXPORT void ob_frame_copy_info ( const ob_frame * src_frame,  
                                    ob_frame *      dst_frame,  
                                    ob_error **     error )
```

Copy the information of the source frame object to the destination frame object.

Including the index, timestamp, system timestamp, global timestamp and metadata will be copied.

#### Parameters

- [in] **src\_frame** Source frame object to copy the information from.
- [in] **dst\_frame** Destination frame object to copy the information to.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### ◆ [ob\\_frame\\_get\\_index\(\)](#)

```
OB_EXPORT uint64_t ob_frame_get_index( const ob_frame * frame,  
                                      ob_error **      error )
```

Get the frame index.

#### Parameters

- [in] **frame** Frame object
- [out] **error** Log wrong message

#### Returns

uint64\_t return the frame index

Referenced by [\*\*ob::Frame::getIndex\(\)\*\*](#).

#### ◆ **ob\_frame\_get\_format()**

```
OB_EXPORT ob_format ob_frame_get_format( const ob_frame * frame,  
                                       ob_error **      error )
```

Get the frame format.

#### Parameters

- [in] **frame** Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_format** return the frame format

Referenced by [\*\*ob::Frame::getFormat\(\)\*\*](#).

#### ◆ **ob\_frame\_get\_type()**

```
OB_EXPORT ob_frame_type ob_frame_get_type ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get the frame type.

#### Parameters

- [in] **frame** Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_frame\_type** return the frame type

Referenced by [ob::Frame::getType\(\)](#).

### ◆ [ob\\_frame\\_get\\_timestamp\\_us\(\)](#)

```
OB_EXPORT uint64_t ob_frame_get_timestamp_us ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get the frame timestamp (also known as device timestamp, hardware timestamp) of the frame in microseconds.

The hardware timestamp is the time point when the frame was captured by the device (Typically in the mid-exposure, unless otherwise stated), on device clock domain.

#### Parameters

- [in] **frame** Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

uint64\_t return the frame hardware timestamp in microseconds

Referenced by [ob::Frame::getTimeStampUs\(\)](#).

### ◆ [ob\\_frame\\_set\\_timestamp\\_us\(\)](#)

```
OB_EXPORT void ob_frame_set_timestamp_us ( ob_frame * frame,  
                                         uint64_t      timestamp_us,  
                                         ob_error ** error )
```

Set the frame timestamp (also known as the device timestamp, hardware timestamp) of a frame object.

#### Parameters

- [in] **frame** Frame object to set the timestamp.
- [in] **timestamp\_us** frame timestamp to set in microseconds.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::FrameHelper::setFrameDeviceTimestampUs\(\)](#).

#### ◆ [ob\\_frame\\_get\\_system\\_timestamp\\_us\(\)](#)

```
OB_EXPORT uint64_t ob_frame_get_system_timestamp_us ( const ob_frame * frame,  
                                                       ob_error ** error )
```

Get the system timestamp of the frame in microseconds.

The system timestamp is the time point when the frame was received by the host, on host clock domain.

#### Parameters

- [in] **frame** Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**uint64\_t** return the frame system timestamp in microseconds

Referenced by [ob::Frame::getSystemTimeStampUs\(\)](#).

#### ◆ [ob\\_frame\\_set\\_system\\_timestamp\\_us\(\)](#)

```
OB_EXPORT void ob_frame_set_system_timestamp_us ( ob_frame * frame,  
                                              uint64_t system_timestamp_us,  
                                              ob_error ** error )
```

Set the system timestamp of the frame in microseconds.

#### Parameters

<b>frame</b>	Frame object
<b>system_timestamp_us</b>	frame system timestamp to set in microseconds.
<b>error</b>	Pointer to an error object that will be set if an error occurs.

#### ◆ **ob\_frame\_get\_global\_timestamp\_us()**

```
OB_EXPORT uint64_t ob_frame_get_global_timestamp_us ( const ob_frame * frame,  
                                                       ob_error ** error )
```

Get the global timestamp of the frame in microseconds.

The global timestamp is the time point when the frame was captured by the device, and has been converted to the host clock domain. The conversion process base on the frame timestamp and can eliminate the timer drift of the device

#### Attention

The global timestamp is disabled by default. If global timestamp is not enabled, the function will return 0. To enable it, call **ob\_device\_enable\_global\_timestamp()** function.

Only some models of device support getting the global timestamp. Check the device support status by **ob\_device\_is\_global\_timestamp\_supported()** function.

#### Parameters

[in] <b>frame</b>	Frame object
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

#### Returns

uint64\_t The global timestamp of the frame in microseconds.

Referenced by **ob::Frame::getGlobalTimeStampUs()**.

#### ◆ **ob\_frame\_get\_data()**

```
OB_EXPORT uint8_t * ob_frame_get_data ( const ob_frame * frame,  
                                         ob_error **      error )
```

Get the data buffer of a frame.

### Attention

The returned data buffer is mutable, but it is not recommended to modify it directly. Modifying the data directly may cause issues if the frame is being used in other threads or future use. If you need to modify the data, it is recommended to create a new frame object.

### Parameters

- [in] **frame** The frame object from which to retrieve the data.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint8\_t\* Pointer to the frame data buffer.

Referenced by **ob::Frame::getData()**.

◆ **ob\_frame\_update\_data()**

```
OB_EXPORT void ob_frame_update_data ( ob_frame * frame,
                                      const uint8_t * data,
                                      uint32_t data_size,
                                      ob_error ** error )
```

Update the data of a frame.

The data will be memcpy to the frame data buffer.

The frame data size will be also updated as the input data size.

### Attention

It is not recommended to update the frame data if the frame was not created by the user. If you must update it, ensure that the frame is not being used in other threads.

The size of the new data should be equal to or less than the current data size of the frame.

Exceeding the original size may cause memory exceptions.

### Parameters

- [in] **frame** The frame object to update.
- [in] **data** The new data to update the frame with.
- [in] **data\_size** The size of the new data.
- [out] **error** Pointer to an error object that will be set if an error occurs.

## ◆ ob\_frame\_get\_data\_size()

```
OB_EXPORT uint32_t ob_frame_get_data_size ( const ob_frame * frame,
                                              ob_error ** error )
```

Get the frame data size.

### Parameters

- [in] **frame** Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint32\_t return the frame data size If it is point cloud data, it return the number of bytes occupied by all point sets. If you need to find the number of points, you need to divide dataSize by the structure size of the corresponding point type.

Referenced by [ob::Frame::getDataSize\(\)](#).

◆ **ob\_frame\_get\_metadata()**

```
OB_EXPORT uint8_t * ob_frame_get_metadata ( const ob_frame * frame,  
                                         ob_error **      error )
```

Get the metadata of the frame.

**Attention**

The returned metadata is mutable, but it is not recommended to modify it directly. Modifying the metadata directly may cause issues if the frame is being used in other threads or future use. If you need to modify the metadata, it is recommended to create a new frame object.

**Parameters**

[in] **frame** frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

**Returns**

const uint8\_t \* return the metadata pointer of the frame

Referenced by **ob::Frame::getMetadata()**.

◆ **ob\_frame\_get\_metadata\_size()**

```
OB_EXPORT uint32_t ob_frame_get_metadata_size ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get the metadata size of the frame.

**Parameters**

[in] **frame** frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

**Returns**

uint32\_t return the metadata size of the frame

Referenced by **ob::Frame::getMetadataSize()**.

◆ **ob\_frame\_update\_metadata()**

```
OB_EXPORT void ob_frame_update_metadata ( ob_frame * frame,  
                                         const uint8_t * metadata,  
                                         uint32_t metadata_size,  
                                         ob_error ** error )
```

Update the metadata of the frame.

The metadata will be memcpy to the frame metadata buffer.

The frame metadata size will be also updated as the input metadata size.

### Attention

It is not recommended to update the frame metadata if the frame was not created by the user. If you must update it, ensure that the frame is not being used in other threads or future use.

The metadata size should be equal to or less than 256 bytes, otherwise it will cause memory exception.

### Parameters

[in] **frame** frame object  
[in] **metadata** The new metadata to update.  
[in] **metadata\_size** The size of the new metadata.  
[out] **error** Pointer to an error object that will be set if an error occurs.

### ◆ **ob\_frame\_has\_metadata()**

```
OB_EXPORT bool ob_frame_has_metadata ( const ob_frame * frame,  
                                         ob_frame_metadata_type type,  
                                         ob_error ** error )
```

check if the frame contains the specified metadata

### Parameters

[in] **frame** frame object  
[in] **type** metadata type, refer to **ob\_frame\_metadata\_type**  
[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Frame::hasMetadata()**.

### ◆ **ob\_frame\_get\_metadata\_value()**

```
OB_EXPORT int64_t ob_frame_get_metadata_value ( const ob_frame * frame,  
                                              ob_frame_metadata_type type,  
                                              ob_error ** error )
```

Get the metadata value of the frame.

#### Parameters

- [in] **frame** frame object
- [in] **type** metadata type, refer to **ob\_frame\_metadata\_type**
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

int64\_t return the metadata value of the frame

Referenced by **ob::Frame::getMetadataValue()**.

- ◆ **ob\_frame\_get\_stream\_profile()**

```
OB_EXPORT ob_stream_profile * ob_frame_get_stream_profile ( const ob_frame * frame,  
                                                       ob_error ** error )
```

Get the stream profile of the frame.

#### Attention

Require **ob\_delete\_stream\_profile()** to release the return stream profile.

#### Parameters

- frame** frame object
- error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_stream\_profile\*** Return the stream profile of the frame, if the frame is not captured by a sensor stream, it will return NULL

Referenced by **ob::Frame::getStreamProfile()**.

- ◆ **ob\_frame\_set\_stream\_profile()**

```
OB_EXPORT void ob_frame_set_stream_profile ( ob_frame * frame,  
                                              const ob_stream_profile * stream_profile,  
                                              ob_error ** error )
```

Set (override) the stream profile of the frame.

#### Parameters

**frame** frame object

**stream\_profile** The stream profile to set for the frame.

**error** Pointer to an error object that will be set if an error occurs.

#### ◆ **ob\_frame\_get\_sensor()**

```
OB_EXPORT ob_sensor * ob_frame_get_sensor ( const ob_frame * frame,  
                                              ob_error ** error )
```

Get the sensor of the frame.

#### Attention

Require **ob\_delete\_sensor()** to release the return sensor.

#### Parameters

[in] **frame** frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_sensor\*** return the sensor of the frame, if the frame is not captured by a sensor or the sensor stream has been destroyed, it will return NULL

Referenced by **ob::Frame::getSensor()**.

#### ◆ **ob\_frame\_get\_device()**

```
OB_EXPORT ob_device * ob_frame_get_device ( const ob_frame * frame,  
                                         ob_error **          error )
```

Get the device of the frame.

### Attention

Require [ob\\_delete\\_device\(\)](#) to release the return device.

### Parameters

**frame** frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_device**\* return the device of the frame, if the frame is not captured by a sensor stream or the device has been destroyed, it will return NULL

Referenced by [ob::Frame::getDevice\(\)](#).

- ◆ [ob\\_video\\_frame\\_get\\_width\(\)](#)

```
OB_EXPORT uint32_t ob_video_frame_get_width ( const ob_frame * frame,  
                                              ob_error **          error )
```

Get video frame width.

### Parameters

[in] **frame** Frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint32\_t return the frame width

Referenced by [ob::VideoFrame::getWidth\(\)](#).

- ◆ [ob\\_video\\_frame\\_get\\_height\(\)](#)

```
OB_EXPORT uint32_t ob_video_frame_get_height ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get video frame height.

#### Parameters

- [in] **frame** Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

uint32\_t return the frame height

Referenced by [ob::VideoFrame::getHeight\(\)](#).

- ◆ [ob\\_video\\_frame\\_get\\_pixel\\_type\(\)](#)

```
OB_EXPORT ob_pixel_type ob_video_frame_get_pixel_type ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get video frame pixel format.

Usually used to determine the pixel type of depth frame (depth, disparity, raw phase, etc.)

#### Attention

Always return OB\_PIXEL\_UNKNOWN for non-depth frame currently if user has not set the pixel type by [ob\\_video\\_frame\\_set\\_pixel\\_type\(\)](#)

#### Parameters

- frame** Frame object
- error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_pixel\_type** return the pixel format of the frame.

Referenced by [ob::VideoFrame::getPixelType\(\)](#).

- ◆ [ob\\_video\\_frame\\_set\\_pixel\\_type\(\)](#)

```
OB_EXPORT void ob_video_frame_set_pixel_type ( ob_frame * frame,  
                                              ob_pixel_type pixel_type,  
                                              ob_error ** error )
```

Set video frame pixel format.

#### Parameters

**frame** Frame object

**pixel\_type** the pixel format of the frame

**error** Pointer to an error object that will be set if an error occurs.

#### ◆ [ob\\_video\\_frame\\_get\\_pixel\\_available\\_bit\\_size\(\)](#)

```
OB_EXPORT uint8_t ob_video_frame_get_pixel_available_bit_size ( const ob_frame * frame,  
                                                               ob_error ** error )
```

Get the effective number of pixels (such as Y16 format frame, but only the lower 10 bits are effective bits, and the upper 6 bits are filled with 0)

#### Attention

Only valid for Y8/Y10/Y11/Y12/Y14/Y16 format

#### Parameters

[in] **frame** video frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

uint8\_t return the effective number of pixels in the pixel, or 0 if it is an unsupported format

Referenced by [ob::VideoFrame::getPixelAvailableBitSize\(\)](#).

#### ◆ [ob\\_video\\_frame\\_set\\_pixel\\_available\\_bit\\_size\(\)](#)

Set the effective number of pixels (such as Y16 format frame, but only the lower 10 bits are effective bits, and the upper 6 bits are filled with 0)

## Attention

Only valid for Y8/Y10/Y11/Y12/Y14/Y16 format

## Parameters

[in] **frame** video frame object

[in] **bit\_size** the effective number of pixels in the pixel, or 0 if it is an unsupported format

[out] **error** Pointer to an error object that will be set if an error occurs.

#### ◆ ob\_ir\_frame\_get\_source\_sensor\_type()

Get the source sensor type of the ir frame (left or right for dual camera)

## Parameters

**frame** Frame object

**ob\_error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_sensor\_type** return the source sensor type of the ir frame

#### ◆ ob\_depth\_frame\_get\_value\_scale()

```
OB_EXPORT float ob_depth_frame_get_value_scale ( const ob_frame * frame,  
                                              ob_error ** error )
```

Get the value scale of the depth frame. The pixel value of the depth frame is multiplied by the scale to give a depth value in millimeters. For example, if valueScale=0.1 and a certain coordinate pixel value is pixelValue=10000, then the depth value = pixelValue\*valueScale = 10000\*0.1=1000mm.

### Parameters

[in] **frame** Frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

float The value scale of the depth frame

Referenced by [ob::DepthFrame::getValueScale\(\)](#).

### ◆ [ob\\_depth\\_frame\\_set\\_value\\_scale\(\)](#)

```
OB_EXPORT void ob_depth_frame_set_value_scale ( ob_frame * frame,  
                                                float      value_scale,  
                                                ob_error ** error )
```

Set the value scale of the depth frame. The pixel value of the depth frame is multiplied by the scale to give a depth value in millimeters. For example, if valueScale=0.1 and a certain coordinate pixel value is pixelValue=10000, then the depth value = pixelValue\*valueScale = 10000\*0.1=1000mm.

### Parameters

[in] **frame** Frame object

[in] **value\_scale** The value scale of the depth frame

[out] **error** Pointer to an error object that will be set if an error occurs.

### ◆ [ob\\_points\\_frame\\_get\\_coordinate\\_value\\_scale\(\)](#)

Get the point coordinate value scale of the points frame. The point position value of the points frame is multiplied by the scale to give a position value in millimeters. For example, if scale=0.1, the x-coordinate value of a point is x = 10000, which means that the actual x-coordinate value =  $x * \text{scale} = 10000 * 0.1 = 1000\text{mm}$ .

## Parameters

[in] **frame** Frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

float The coordinate value scale of the points frame

Referenced by [ob::PointsFrame::getCoordinateValueScale\(\)](#).

- #### ◆ ob\_accel\_frame\_get\_value()

Get accelerometer frame data.

## Parameters

[in] **frame** Accelerometer frame.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_accel\_value** Return the accelerometer data.

Referenced by [ob::AccelFrame::getValue\(\)](#).

- #### ◆ ob\_accel\_frame\_get\_temperature()

```
OB_EXPORT float ob_accel_frame_get_temperature ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get the temperature when acquiring the accelerometer frame.

### Parameters

- [in] **frame** Accelerometer frame.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

float Return the temperature value.

Referenced by [ob::AccelFrame::getTemperature\(\)](#).

### ◆ [ob\\_gyro\\_frame\\_get\\_value\(\)](#)

```
OB_EXPORT ob_gyro_value ob_gyro_frame_get_value ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get gyroscope frame data.

### Parameters

- [in] **frame** Gyroscope frame.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_gyro\_value** Return the gyroscope data.

Referenced by [ob::GyroFrame::getValue\(\)](#).

### ◆ [ob\\_gyro\\_frame\\_get\\_temperature\(\)](#)

```
OB_EXPORT float ob_gyro_frame_get_temperature ( const ob_frame * frame,  
                                              ob_error **      error )
```

Get the temperature when acquiring the gyroscope frame.

### Parameters

- [in] **frame** Gyroscope frame.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

float Return the temperature value.

Referenced by [ob::GyroFrame::getTemperature\(\)](#).

### ◆ [ob\\_frameset\\_get\\_count\(\)](#)

```
OB_EXPORT uint32_t ob_frameset_get_count ( const ob_frame * frameset,  
                                         ob_error **      error )
```

Get the number of frames contained in the frameset.

### Attention

The frame returned by this function should call [ob\\_delete\\_frame\(\)](#) to decrease the reference count when it is no longer needed.

### Parameters

- [in] **frameset** frameset object
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint32\_t return the number of frames

Referenced by [ob::FrameSet::getCount\(\)](#).

### ◆ [ob\\_frameset\\_get\\_depth\\_frame\(\)](#)

```
OB_EXPORT ob_frame * ob_frameset_get_depth_frame ( const ob_frame * frameset,  
                                              ob_error **      error )
```

Get the depth frame from the frameset.

### Attention

The frame returned by this function should call **ob\_delete\_frame()** to decrease the reference count when it is no longer needed.

### Parameters

[in] **frameset** Frameset object.

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_frame\*** Return the depth frame.

## ◆ **ob\_frameset\_get\_color\_frame()**

```
OB_EXPORT ob_frame * ob_frameset_get_color_frame ( const ob_frame * frameset,  
                                              ob_error **      error )
```

Get the color frame from the frameset.

### Attention

The frame returned by this function should call **ob\_delete\_frame()** to decrease the reference count when it is no longer needed.

### Parameters

[in] **frameset** Frameset object.

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_frame\*** Return the color frame.

## ◆ **ob\_frameset\_get\_ir\_frame()**

```
OB_EXPORT ob_frame * ob_frameset_get_ir_frame ( const ob_frame * frameset,  
                                              ob_error **      error )
```

Get the infrared frame from the frameset.

### Attention

The frame returned by this function should call **ob\_delete\_frame()** to decrease the reference count when it is no longer needed.

### Parameters

- [in] **frameset** Frameset object.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

ob\_frame\* Return the infrared frame.

## ◆ ob\_frameset\_get\_points\_frame()

```
OB_EXPORT ob_frame * ob_frameset_get_points_frame ( const ob_frame * frameset,  
                                              ob_error **      error )
```

Get point cloud frame from the frameset.

### Attention

The frame returned by this function should call **ob\_delete\_frame()** to decrease the reference count when it is no longer needed.

### Parameters

- [in] **frameset** Frameset object.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

ob\_frame\* Return the point cloud frame.

## ◆ ob\_frameset\_get\_frame()

```
OB_EXPORT ob_frame * ob_frameset_get_frame ( const ob_frame * frameset,  
                                              ob_frame_type    frame_type,  
                                              ob_error **      error )
```

Get a frame of a specific type from the frameset.

### Attention

The frame returned by this function should call **ob\_delete\_frame()** to decrease the reference count when it is no longer needed.

### Parameters

[in] **frameset** Frameset object.  
[in] **frame\_type** Frame type.  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_frame\*** Return the frame of the specified type, or nullptr if it does not exist.

Referenced by **ob::FrameSet::getFrame()**.

### ◆ ob\_frameset\_get\_frame\_by\_index()

```
OB_EXPORT ob_frame * ob_frameset_get_frame_by_index ( const ob_frame * frameset,  
                                                       uint32_t          index,  
                                                       ob_error **      error )
```

Get a frame at a specific index from the FrameSet.

### Parameters

[in] **frameset** Frameset object.  
[in] **index** The index of the frame.  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_frame\*** Return the frame at the specified index, or nullptr if it does not exist.

Referenced by **ob::FrameSet::getFrameByIndex()**.

### ◆ ob\_frameset\_push\_frame()

```
OB_EXPORT void ob_frameset_push_frame ( ob_frame * frameset,  
                                         const ob_frame * frame,  
                                         ob_error ** error )
```

Push a frame to the frameset.

### Attention

If a frame with same type already exists in the frameset, it will be replaced by the new frame.

The frame push to the frameset will be add reference count, so you still need to call

[\*\*ob\\_delete\\_frame\(\)\*\*](#) to decrease the reference count when it is no longer needed.

### Parameters

[in] **frameset** Frameset object.

[in] **frame** Frame object to push.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [\*\*ob::FrameSet::pushFrame\(\)\*\*](#).

### ◆ [\*\*ob\\_point\\_cloud\\_frame\\_get\\_width\(\)\*\*](#)

```
OB_EXPORT uint32_t ob_point_cloud_frame_get_width ( const ob_frame * frame,  
                                                       ob_error ** error )
```

Get point cloud frame width.

### Parameters

[in] **frame** point cloud Frame object

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint32\_t return the point cloud frame width

Referenced by [\*\*ob::PointsFrame::getWidth\(\)\*\*](#).

### ◆ [\*\*ob\\_point\\_cloud\\_frame\\_get\\_height\(\)\*\*](#)

```
OB_EXPORT uint32_t ob_point_cloud_frame_get_height ( const ob_frame * frame,  
                                                 ob_error **      error )
```

Get point cloud frame height.

### Parameters

- [in] **frame** point cloud Frame object
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint32\_t return the point cloud frame height

Referenced by [ob::PointsFrame::getHeight\(\)](#).

# Frame.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6 #ifndef __cplusplus
7 extern "C" {
8#endif
9
10 #include "ObTypes.h"
11
12 OB_EXPORT ob_frame *ob_create_frame(ob_frame_type frame_type, ob_format format, uint32_t data_size,
13                                     ob_error **error);
14
15 OB_EXPORT ob_frame *ob_create_frame_from_other_frame(const ob_frame *other_frame, bool should_copy_data,
16                                                       ob_error **error);
17
18 OB_EXPORT ob_frame *ob_create_frame_from_stream_profile(const ob_stream_profile *stream_profile, ob_error **error);
19
20 OB_EXPORT ob_frame *ob_create_video_frame(ob_frame_type frame_type, ob_format format, uint32_t width, uint32_t height, uint32_t stride_bytes, ob_error **error);
21
22 OB_EXPORT ob_frame *ob_create_frame_from_buffer(ob_frame_type frame_type, ob_format format, uint8_t *buffer, uint32_t buffer_size, ob_frame_destroy_callback *buffer_destroy_cb, void *buffer_destroy_context, ob_error **error);
23
24 OB_EXPORT ob_frame *ob_create_video_frame_from_buffer(ob_frame_type frame_type, ob_format format, uint32_t width, uint32_t height, uint32_t stride_bytes, uint8_t *buffer, uint32_t buffer_size, ob_frame_destroy_callback *buffer_destroy_cb, void *buffer_destroy_context, ob_error **error);
25
26 OB_EXPORT ob_frame *ob_create_frameset(ob_error **error);
27
28 OB_EXPORT void ob_frame_add_ref(const ob_frame *frame, ob_error **error);
29
30 OB_EXPORT void ob_delete_frame(const ob_frame *frame, ob_error **error);
31
32 OB_EXPORT void ob_frame_copy_info(const ob_frame *src_frame, ob_frame *dst_frame, ob_error **error);
33
34 OB_EXPORT uint64_t ob_frame_get_index(const ob_frame *frame, ob_error **error);
35
36 OB_EXPORT ob_format ob_frame_get_format(const ob_frame *frame, ob_error **error);
37
38 OB_EXPORT ob_frame_type ob_frame_get_type(const ob_frame *frame, ob_error **error);
39
40 OB_EXPORT uint64_t ob_frame_get_timestamp_us(const ob_frame *frame, ob_error **error);
41
42 OB_EXPORT void ob_frame_set_timestamp_us(ob_frame *frame, uint64_t timestamp_us, ob_error **error);
43
44 OB_EXPORT uint64_t ob_frame_get_system_timestamp_us(const ob_frame *frame, ob_error **error);
45
46 OB_EXPORT void ob_frame_set_system_timestamp_us(ob_frame *frame, uint64_t system_timestamp_us, ob_error **error);
```

```

224
239 OB_EXPORT uint64_t ob_frame_get_global_timestamp_us(const ob_frame *frame, ob_error **error);
240
251 OB_EXPORT uint8_t *ob_frame_get_data(const ob_frame *frame, ob_error **error);
252
268 OB_EXPORT void ob_frame_update_data(ob_frame *frame, const uint8_t *data, uint32_t data_size, ob_error **error);
269
279 OB_EXPORT uint32_t ob_frame_get_data_size(const ob_frame *frame, ob_error **error);
280
291 OB_EXPORT uint8_t *ob_frame_get_metadata(const ob_frame *frame, ob_error **error);
292 #define ob_video_frame_metadata ob_frame_get_metadata // for compatibility
293
301 OB_EXPORT uint32_t ob_frame_get_metadata_size(const ob_frame *frame, ob_error **error);
302 #define ob_video_frame_metadata_size ob_frame_get_metadata_size // for compatibility
303
318 OB_EXPORT void ob_frame_update_metadata(ob_frame *frame, const uint8_t *metadata, uint32_t metadata_size, ob_error **error);
319
327 OB_EXPORT bool ob_frame_has_metadata(const ob_frame *frame, ob_frame_metadata_type type, ob_error **error);
328
337 OB_EXPORT int64_t ob_frame_get_metadata_value(const ob_frame *frame, ob_frame_metadata_type type, ob_error **error);
338
348 OB_EXPORT ob_stream_profile *ob_frame_get_stream_profile(const ob_frame *frame, ob_error **error);
349
357 OB_EXPORT void ob_frame_set_stream_profile(ob_frame *frame, const ob_stream_profile *stream_profile, ob_error **error);
358
368 OB_EXPORT ob_sensor *ob_frame_get_sensor(const ob_frame *frame, ob_error **error);
369
379 OB_EXPORT ob_device *ob_frame_get_device(const ob_frame *frame, ob_error **error);
380
388 OB_EXPORT uint32_t ob_video_frame_get_width(const ob_frame *frame, ob_error **error);
389
397 OB_EXPORT uint32_t ob_video_frame_get_height(const ob_frame *frame, ob_error **error);
398
409 OB_EXPORT ob_pixel_type ob_video_frame_get_pixel_type(const ob_frame *frame, ob_error **error);
410
418 OB_EXPORT void ob_video_frame_set_pixel_type(ob_frame *frame, ob_pixel_type pixel_type, ob_error **error);
419
428 OB_EXPORT uint8_t ob_video_frame_get_pixel_available_bit_size(const ob_frame *frame, ob_error **error);
429
438 OB_EXPORT void ob_video_frame_set_pixel_available_bit_size(ob_frame *frame, uint8_t bit_size, ob_error **error);
439
447 OB_EXPORT ob_sensor_type ob_ir_frame_get_source_sensor_type(const ob_frame *frame, ob_error **error);
448
457 OB_EXPORT float ob_depth_frame_get_value_scale(const ob_frame *frame, ob_error **error);
458
467 OB_EXPORT void ob_depth_frame_set_value_scale(ob_frame *frame, float value_scale, ob_error **error);
468
478 OB_EXPORT float ob_points_frame_get_coordinate_value_scale(const ob_frame *frame, ob_error **error);
479
487 OB_EXPORT ob_accel_value ob_accel_frame_get_value(const ob_frame *frame, ob_error **error);
488
496 OB_EXPORT float ob_accel_frame_get_temperature(const ob_frame *frame, ob_error **error);
497
505 OB_EXPORT ob_gyro_value ob_gyro_frame_get_value(const ob_frame *frame, ob_error **error);

```

```

506
514 OB_EXPORT float ob_gyro_frame_get_temperature(const ob_frame *frame, ob_error **error);
515
525 OB_EXPORT uint32_t ob_frameset_get_count(const ob_frame *frameset, ob_error **error);
526
536 OB_EXPORT ob_frame *ob_frameset_get_depth_frame(const ob_frame *frameset, ob_error **error);
537
547 OB_EXPORT ob_frame *ob_frameset_get_color_frame(const ob_frame *frameset, ob_error **error);
548
558 OB_EXPORT ob_frame *ob_frameset_get_ir_frame(const ob_frame *frameset, ob_error **error);
559
569 OB_EXPORT ob_frame *ob_frameset_get_points_frame(const ob_frame *frameset, ob_error **error);
570
581 OB_EXPORT ob_frame *ob_frameset_get_frame(const ob_frame *frameset, ob_frame_type frame_type, o
      b_error **error);
582
591 OB_EXPORT ob_frame *ob_frameset_get_frame_by_index(const ob_frame *frameset, uint32_t index, ob_
      error **error);
592
604 OB_EXPORT void ob_frameset_push_frame(ob_frame *frameset, const ob_frame *frame, ob_error **erro
      r);
605
613 OB_EXPORT uint32_t ob_point_cloud_frame_get_width(const ob_frame *frame, ob_error **error);
614
622 OB_EXPORT uint32_t ob_point_cloud_frame_get_height(const ob_frame *frame, ob_error **error);
623
624 // The following interfaces are deprecated and are retained here for compatibility purposes.
625 #define ob_frame_index ob_frame_get_index
626 #define ob_frame_format ob_frame_get_format
627 #define ob_frame_time_stamp_us ob_frame_get_timestamp_us
628 #define ob_frame_set_device_time_stamp_us ob_frame_set_timestamp_us
629 #define ob_frame_system_time_stamp_us ob_frame_get_system_timestamp_us
630 #define ob_frame_global_time_stamp_us ob_frame_get_global_timestamp_us
631 #define ob_frame_data ob_frame_get_data
632 #define ob_frame_data_size ob_frame_get_data_size
633 #define ob_frame_metadata ob_frame_get_metadata
634 #define ob_frame_metadata_size ob_frame_get_metadata_size
635 #define ob_video_frame_width ob_video_frame_get_width
636 #define ob_video_frame_height ob_video_frame_get_height
637 #define ob_video_frame_pixel_available_bit_size ob_video_frame_get_pixel_available_bit_size
638 #define ob_points_frame_get_position_value_scale ob_points_frame_get_coordinate_value_scale
639 #define ob_frameset_frame_count ob_frameset_get_count
640 #define ob_frameset_depth_frame ob_frameset_get_depth_frame
641 #define ob_frameset_color_frame ob_frameset_get_color_frame
642 #define ob_frameset_ir_frame ob_frameset_get_ir_frame
643 #define ob_frameset_points_frame ob_frameset_get_points_frame
644 #define ob_accel_frame_value ob_accel_frame_get_value
645 #define ob_accel_frame_temperature ob_accel_frame_get_temperature
646 #define ob_gyro_frame_value ob_gyro_frame_get_value
647 #define ob_gyro_frame_temperature ob_gyro_frame_get_temperature
648 #define ob_frameset_get_frame_count ob_frameset_get_count
649
650 #define ob_frame_time_stamp(frame, err) (ob_frame_get_timestamp_us(frame, err) / 1000)
651 #define ob_frame_system_time_stamp(frame, err) (ob_frame_get_system_timestamp_us(frame, err))
652 #define ob_frame_set_system_time_stamp(frame, system_timestamp, err) (ob_frame_set_system_timestamp_
      us(frame, system_timestamp * 1000, err))
653 #define ob_frame_set_device_time_stamp(frame, device_timestamp, err) (ob_frame_set_timestamp_us(fra
      me, device_timestamp * 1000, err))
654
655 #ifdef __cplusplus
656 }
657#endif

```



# Frame.hpp File Reference

Frame related type, which is mainly used to obtain frame data and frame information. [More...](#)

```
#include "Types.hpp"
#include "libobsensor/h/Frame.h"
#include "libobsensor/hpp/Error.hpp"
#include "libobsensor/hpp/StreamProfile.hpp"
#include <memory>
#include <iostream>
#include <typeinfo>
#include <functional>
```

[Go to the source code of this file.](#)

## Classes

class **ob::Frame**

Define the frame class, which is the base class of all frame types. [More...](#)

class **ob::VideoFrame**

Define the **VideoFrame** class, which inherits from the **Frame** class. [More...](#)

class **ob::ColorFrame**

Define the **ColorFrame** class, which inherits from the **VideoFrame** classd. [More...](#)

class **ob::DepthFrame**

Define the **DepthFrame** class, which inherits from the **VideoFrame** class. [More...](#)

class **ob::IRFrame**

Define the **IRFrame** class, which inherits from the **VideoFrame** class. [More...](#)

class **ob::ConfidenceFrame**

Define the **ConfidenceFrame** class, which inherits from the **VideoFrame** class. [More...](#)

class **ob::PointsFrame**

Define the **PointsFrame** class, which inherits from the **Frame** class. [More...](#)

class **ob::AccelFrame**

Define the **AccelFrame** class, which inherits from the **Frame** class. [More...](#)

class **ob::GyroFrame**

Define the **GyroFrame** class, which inherits from the **Frame** class. [More...](#)

class **ob::FrameSet**

Define the **FrameSet** class, which inherits from the **Frame** class. [More...](#)

class **ob::FrameFactory**

**FrameFactory** class, which provides some static functions to create frame objects. [More...](#)

## class **ob::FrameHelper**

FrameHepler class, which provides some static functions to set timestamp for frame objects  
FrameHepler inherited from the **FrameFactory** and the timestamp interface implement here both  
for compatibility purposes. [More...](#)

## Namespaces

namespace **ob**

## Macros

```
#define getPositionValueScale getCoordinateValueScale
```

## Detailed Description

Frame related type, which is mainly used to obtain frame data and frame information.

Definition in file [Frame.hpp](#).

## Macro Definition Documentation

### ◆ **getPositionValueScale**

```
#define getPositionValueScale getCoordinateValueScale
```

Definition at line [647](#) of file [Frame.hpp](#).

# Frame.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6
7 namespace ob {
8 class Device;
9 class Sensor;
10
11 class Frame : public std::enable_shared_from_this<Frame> {
12 protected:
13     const ob_frame *impl_ = nullptr;
14
15 public:
16     explicit Frame(const ob_frame *impl) : impl_(impl) {}
17
18     const ob_frame *getImpl() const {
19         return impl_;
20     }
21
22     virtual ~Frame() noexcept {
23         if(impl_) {
24             ob_error *error = nullptr;
25             ob_delete_frame(impl_, &error);
26             Error::handle(&error, false);
27             impl_ = nullptr;
28         }
29     }
30
31     virtual OBFrameType getType() const {
32         ob_error *error = nullptr;
33         auto type = ob_frame_get_type(impl_, &error);
34         Error::handle(&error);
35
36         return type;
37     }
38
39     virtual OBFormat getFormat() const {
40         ob_error *error = nullptr;
41         auto format = ob_frame_get_format(impl_, &error);
42         Error::handle(&error);
43
44         return format;
45     }
46
47     virtual uint64_t getIndex() const {
48         ob_error *error = nullptr;
49         auto index = ob_frame_get_index(impl_, &error);
50
51         return index;
52     }
53 }
```

```

121     Error::handle(&error);
122
123     return index;
124 }
125
131     virtual uint8_t *getData() const {
132         ob_error *error=nullptr;
133         auto    data = ob_frame_get_data(impl_, &error);
134         Error::handle(&error);
135
136         return data;
137     }
138
146     virtual uint32_t getDataSize() const {
147         ob_error *error =nullptr;
148         auto   dataSize = ob_frame_get_data_size(impl_, &error);
149         Error::handle(&error);
150
151         return dataSize;
152     }
153
160     uint64_t getTimeStampUs() const {
161         ob_error *error =nullptr;
162         auto   timeStampUs = ob_frame_get_timestamp_us(impl_, &error);
163         Error::handle(&error);
164
165         return timeStampUs;
166     }
167
174     uint64_t getSystemTimeStampUs() const {
175         ob_error *error =nullptr;
176         auto   systemTimeStampUs = ob_frame_get_system_timestamp_us(impl_, &error);
177         Error::handle(&error);
178
179         return systemTimeStampUs;
180     }
181
193     uint64_t getGlobalTimeStampUs() const {
194         ob_error *error =nullptr;
195         auto   globalTimeStampUs = ob_frame_get_global_timestamp_us(impl_, &error);
196         Error::handle(&error);
197
198         return globalTimeStampUs;
199     }
200
206     uint8_t *getMetadata() const {
207         ob_error *error =nullptr;
208         auto   metadata = ob_frame_get_metadata(impl_, &error);
209         Error::handle(&error);
210
211         return metadata;
212     }
213
219     uint32_t getMetadataSize() const {
220         ob_error *error =nullptr;
221         auto   metadataSize = ob_frame_get_metadata_size(impl_, &error);
222         Error::handle(&error);
223
224         return metadataSize;
225     }
226
233     bool hasMetadata(OBFrameMetadataType type) const {
234         ob_error *error =nullptr;
235         auto   result = ob_frame_has_metadata(impl_, type, &error);
236         Error::handle(&error);

```

```

237     return result;
238 }
239
240 int64_t getMetadataValue(OBFrameMetadataType type) const {
241     ob_error *error=nullptr;
242     auto    value=ob_frame_get_metadata_value(impl_, type, &error);
243     Error::handle(&error);
244
245     return value;
246 }
247
248 std::shared_ptr<StreamProfile> getStreamProfile() const {
249     ob_error *error =nullptr;
250     auto    profile=ob_frame_get_stream_profile(impl_, &error);
251     Error::handle(&error);
252     return StreamProfileFactory::create(profile);
253 }
254
255 std::shared_ptr<Sensor> getSensor() const {
256     ob_error *error =nullptr;
257     auto    sensor=ob_frame_get_sensor(impl_, &error);
258     Error::handle(&error);
259
260     return std::make_shared<Sensor>(sensor);
261 }
262
263 std::shared_ptr<Device> getDevice() const {
264     ob_error *error =nullptr;
265     auto    device=ob_frame_get_device(impl_, &error);
266     Error::handle(&error);
267
268     return std::make_shared<Device>(device);
269 }
270
271 template <typename T> bool is() const;
272
273 template <typename T> std::shared_ptr<T> as() {
274     if(!is<T>()) {
275         throw std::runtime_error("unsupported operation, object's type is not require type");
276     }
277
278     ob_error *error=nullptr;
279     ob_frame_add_ref(impl_, &error);
280     Error::handle(&error);
281
282     return std::make_shared<T>(impl_);
283 }
284
285 template <typename T> std::shared_ptr<const T> as() const {
286     if(!is<const T>()) {
287         throw std::runtime_error("unsupported operation, object's type is not require type");
288     }
289
290     ob_error *error=nullptr;
291     ob_frame_add_ref(impl_, &error);
292     Error::handle(&error);
293
294     return std::make_shared<const T>(impl_);
295 }
296
297 public:
298     // The following interfaces are deprecated and are retained here for compatibility purposes.
299     OBFrameType type() const {
300         return getType();
301     }

```

```

341     }
342
343     virtual OBFormat format() const {
344         return getFormat();
345     }
346
347     virtual uint64_t index() const {
348         return getIndex();
349     }
350
351     virtual void *data() const {
352         auto data = getData();
353         return reinterpret_cast<void *>(data);
354     }
355
356     virtual uint32_t dataSize() const {
357         return getDataSize();
358     }
359
360     uint64_t timeStamp() const {
361         return getTimeStampUs() / 1000;
362     }
363
364     uint64_t timeStampUs() const {
365         return getTimeStampUs();
366     }
367
368     uint64_t systemTimeStamp() const {
369         return getSystemTimeStampUs() / 1000;
370     }
371
372     uint64_t systemTimeStampUs() const {
373         return getSystemTimeStampUs();
374     }
375
376     uint64_t globalTimeStampUs() const {
377         return getGlobalTimeStampUs();
378     }
379
380     uint8_t *metadata() const {
381         return getMetadata();
382     }
383
384     uint32_t metadataSize() const {
385         return getMetadataSize();
386     }
387 };
388
392 class VideoFrame : public Frame {
393 public:
403     explicit VideoFrame(const ob_frame *impl) : Frame(impl) {}
404
405     ~VideoFrame() noexcept override = default;
406
412     uint32_t getWidth() const {
413         ob_error *error = nullptr;
414         auto width = ob_video_frame_get_width(impl_, &error);
415         Error::handle(&error);
416
417         return width;
418     }
419
425     uint32_t getHeight() const {
426         ob_error *error = nullptr;
427         auto height = ob_video_frame_get_height(impl_, &error);
428         Error::handle(&error);

```

```

429     return height;
430 }
431
432 OBPixelType getPixelType() const {
433     ob_error *error = nullptr;
434     auto pixelType = ob_video_frame_get_pixel_type(impl_, &error);
435     Error::handle(&error);
436
437     return pixelType;
438 }
439
440 uint8_t getPixelAvailableBitSize() const {
441     ob_error *error = nullptr;
442     auto bitSize = ob_video_frame_get_pixel_available_bit_size(impl_, &error);
443     Error::handle(&error);
444
445     return bitSize;
446 }
447
448 public:
449 // The following interfaces are deprecated and are retained here for compatibility purposes.
450     uint32_t width() const {
451         return getWidth();
452     }
453
454     uint32_t height() const {
455         return getHeight();
456     }
457
458     uint8_t pixelAvailableBitSize() const {
459         return getPixelAvailableBitSize();
460     }
461 };
462
463 class ColorFrame : public VideoFrame {
464 public:
465     explicit ColorFrame(const ob_frame *impl) : VideoFrame(impl) {};
466
467     ~ColorFrame() noexcept override = default;
468 };
469
470 class DepthFrame : public VideoFrame {
471 public:
472     explicit DepthFrame(const ob_frame *impl) : VideoFrame(impl) {};
473
474     ~DepthFrame() noexcept override = default;
475
476     float getValueScale() const {
477         ob_error *error = nullptr;
478         auto scale = ob_depth_frame_get_value_scale(impl_, &error);
479         Error::handle(&error);
480
481         return scale;
482     }
483 };
484
485 class IRFrame : public VideoFrame {
486 public:
487     explicit IRFrame(const ob_frame *impl) : VideoFrame(impl) {};
488
489     ~IRFrame() noexcept override = default;
490 };

```

```

555
560 class ConfidenceFrame : public VideoFrame {
561
562 public:
563     explicit ConfidenceFrame(const ob_frame *impl) : VideoFrame(impl){};
564
565     ~ConfidenceFrame() noexcept override = default;
566 };
567
568 class PointsFrame : public Frame {
569
570 public:
571     explicit PointsFrame(const ob_frame *impl) : Frame(impl) {};
572
573     ~PointsFrame() noexcept override = default;
574
575     float getCoordinateValueScale() const {
576         ob_error *error=nullptr;
577         auto scale = ob_points_frame_get_coordinate_value_scale(impl_, &error);
578         Error::handle(&error);
579
580         return scale;
581     }
582
583     uint32_t getWidth() const {
584         ob_error *error=nullptr;
585         // TODO
586         auto width = ob_point_cloud_frame_get_width(impl_, &error);
587         Error::handle(&error);
588
589         return width;
590     }
591
592     uint32_t getHeight() const {
593         ob_error *error =nullptr;
594         auto height = ob_point_cloud_frame_get_height(impl_, &error);
595         Error::handle(&error);
596
597         return height;
598     }
599
600 public:
601     // The following interfaces are deprecated and are retained here for compatibility purposes.
602     #define getPositionValueScale getCoordinateValueScale
603 };
604
605 class AccelFrame : public Frame {
606
607 public:
608     explicit AccelFrame(const ob_frame *impl) : Frame(impl) {};
609
610     ~AccelFrame() noexcept override = default;
611
612     OBAccelValue getValue() const {
613         ob_error *error=nullptr;
614         auto value = ob_accel_frame_get_value(impl_, &error);
615         Error::handle(&error);
616
617         return value;
618     }
619
620     float getTemperature() const {
621         ob_error *error=nullptr;
622         auto temp = ob_accel_frame_get_temperature(impl_, &error);
623         Error::handle(&error);
624
625     }

```

```

683     return temp;
684 }
685
686
687 public:
688 // The following interfaces are deprecated and are retained here for compatibility purposes.
689 OBAccelValue value() {
690     return getValue();
691 }
692
693 float temperature() {
694     return getTemperature();
695 }
696 };
697
701 class GyroFrame : public Frame {
702
703 public:
704     explicit GyroFrame(const ob_frame *impl) : Frame(impl) {};
705
706     ~GyroFrame() noexcept override = default;
707
713     OBGyroValue getValue() const {
714         ob_error *error = nullptr;
715         auto    value = ob_gyro_frame_get_value(impl_, &error);
716         Error::handle(&error);
717
718         return value;
719     }
720
726     float getTemperature() const {
727         ob_error *error    = nullptr;
728         auto    temperature = ob_gyro_frame_get_temperature(impl_, &error);
729         Error::handle(&error);
730
731         return temperature;
732     }
733
734 public:
735 // The following interfaces are deprecated and are retained here for compatibility purposes.
736     OBGyroValue value() {
737         return getValue();
738     }
739
740     float temperature() {
741         return getTemperature();
742     }
743 };
744
749 class FrameSet : public Frame {
750
751 public:
752     explicit FrameSet(const ob_frame *impl) : Frame(impl) {};
753
754     ~FrameSet() noexcept override = default;
755
761     uint32_t getCount() const {
762         ob_error *error = nullptr;
763         auto    count = ob_frameset_get_count(impl_, &error);
764         Error::handle(&error);
765         return count;
766     }
767
774     std::shared_ptr<Frame> getFrame(OBFrameType frameType) const {
775         ob_error *error = nullptr;
776         auto    frame = ob_frameset_get_frame(impl_, frameType, &error);

```

```

//>     name = ob_frameset_get_name(impl_, name_type, &error),
777     if(!frame) {
778         return nullptr;
779     }
780     Error::handle(&error);
781     return std::make_shared<Frame>(frame);
782 }
783
790 std::shared_ptr<Frame> getFrameByIndex(uint32_t index) const {
791     ob_error *error = nullptr;
792     auto frame = ob_frameset_get_frame_by_index(impl_, index, &error);
793     if(!frame) {
794         return nullptr;
795     }
796     Error::handle(&error);
797     return std::make_shared<Frame>(frame);
798 }
799
807 void pushFrame(std::shared_ptr<const Frame> frame) const {
808     ob_error *error = nullptr;
809
810     // unsafe operation, need to cast const to non-const
811     auto unConstImpl = const_cast<ob_frame *>(impl_);
812
813     auto otherImpl = frame->getImpl();
814     ob_frameset_push_frame(unConstImpl, otherImpl, &error);
815
816     Error::handle(&error);
817 }
818
819 public:
820     // The following interfaces are deprecated and are retained here for compatibility purposes.
821     uint32_t frameCount() const {
822         return getCount();
823     }
824
825     std::shared_ptr<DepthFrame> depthFrame() const {
826         auto frame = getFrame(OB_FRAME_DEPTH);
827         if(frame == nullptr) {
828             return nullptr;
829         }
830         auto depthFrame = frame->as<ob::DepthFrame>();
831         return depthFrame;
832     }
833
834     std::shared_ptr<ColorFrame> colorFrame() const {
835         auto frame = getFrame(OB_FRAME_COLOR);
836         if(frame == nullptr) {
837             return nullptr;
838         }
839         auto colorFrame = frame->as<ob::ColorFrame>();
840         return colorFrame;
841     }
842
843     std::shared_ptr<IRFrame> irFrame() const {
844         auto frame = getFrame(OB_FRAME_IR);
845         if(frame == nullptr) {
846             return nullptr;
847         }
848         auto irFrame = frame->as<ob::IRFrame>();
849         return irFrame;
850     }
851
852 public:
853     // The following interfaces are deprecated and are retained here for compatibility purposes.
854     std::shared_ptr<PointsFrame> pointsFrame() const {

```

```

855     auto frame = getFrame(OB_FRAME_POINTS);
856     if(frame == nullptr) {
857         return nullptr;
858     }
859     auto pointsFrame = frame->as<ob::PointsFrame>();
860     return pointsFrame;
861 }
862
863 std::shared_ptr<Frame> getFrame(int index) const {
864     return getFrameByIndex(index);
865 }
866 };
867
871 class FrameFactory {
872 public:
881     static std::shared_ptr<Frame> createFrame(OBFrameType frameType, OBFormat format, uint32_t dataSize
882     e) {
883         ob_error *error = nullptr;
884         auto impl = ob_create_frame(frameType, format, dataSize, &error);
885         Error::handle(&error);
886
887         return std::make_shared<Frame>(impl);
888     }
890
901     static std::shared_ptr<VideoFrame> createVideoFrame(OBFrameType frameType, OBFormat format, uint
902         32_t width, uint32_t height, uint32_t stride = 0) {
903         ob_error *error = nullptr;
904         auto impl = ob_create_video_frame(frameType, format, width, height, stride, &error);
905         Error::handle(&error);
906
907         auto frame = std::make_shared<Frame>(impl);
908         return frame->as<VideoFrame>();
909     }
910
919     static std::shared_ptr<Frame> createFrameFromOtherFrame(std::shared_ptr<const Frame> otherFrame, b
920         ool shouldCopyData = true) {
921         ob_error *error = nullptr;
922         auto otherImpl = otherFrame->getImpl();
923         auto impl = ob_create_frame_from_other_frame(otherImpl, shouldCopyData, &error);
924         Error::handle(&error);
925
926         return std::make_shared<Frame>(impl);
927     }
928
935     static std::shared_ptr<Frame> createFrameFromStreamProfile(std::shared_ptr<const StreamProfile> profil
936         e) {
937         ob_error *error = nullptr;
938         auto impl = ob_create_frame_from_stream_profile(profile->getImpl(), &error);
939         Error::handle(&error);
940
941         return std::make_shared<Frame>(impl);
942     }
943
946     typedef std::function<void(uint8_t *)> BufferDestroyCallback;
947
962     static std::shared_ptr<Frame> createFrameFromBuffer(OBFrameType frameType, OBFormat format, uint
963         8_t *buffer, BufferDestroyCallback destroyCallback,
964         uint32_t bufferSize) {
965         ob_error *error = nullptr;
966         auto ctx = new BufferDestroyContext{destroyCallback};
967         auto impl = ob_create_frame_from_buffer(frameType, format, buffer, bufferSize, &FrameFactory::B
968         ufferDestroy, ctx, &error);
969         Error::handle(&error);
970
971         return std::make_shared<Frame>(impl);

```

```

970 }
971
972 static std::shared_ptr<VideoFrame> createVideoFrameFromBuffer(OBFrameType frameType, OBFormat f
973 ormat, uint32_t width, uint32_t height, uint8_t *buffer,
974                                         BufferDestroyCallback destroyCallback, uint32_t bufferSize, uint32_t
975 stride = 0) {
976     ob_error *error = nullptr;
977     auto ctx = new BufferDestroyContext{destroyCallback};
978     auto impl = ob_create_video_frame_from_buffer(frameType, format, width, height, buffer, bufferSize, &FrameFactory::BufferDestroy, ctx, &error);
979     Error::handle(&error);
980
981     auto frame = std::make_shared<Frame>(impl);
982     return frame->as<VideoFrame>();
983 }
984
985 private:
986     struct BufferDestroyContext {
987         BufferDestroyCallback callback;
988     };
989
990     static void BufferDestroy(uint8_t *buffer, void *context) {
991         auto *ctx = static_cast<BufferDestroyContext *>(context);
992         if(ctx->callback) {
993             ctx->callback(buffer);
994         }
995         delete ctx;
996     }
997 };
998
999 class FrameHelper : public FrameFactory {
1000 public:
1001     static void setFrameDeviceTimestampUs(std::shared_ptr<Frame> frame, uint64_t deviceTimestampUs)
1002     {
1003         ob_error *error = nullptr;
1004         auto impl = const_cast<ob_frame *>(frame->getImpl());
1005         ob_frame_set_timestamp_us(impl, deviceTimestampUs, &error);
1006         Error::handle(&error);
1007     }
1008
1009 public:
1010     // The following interfaces are deprecated and are retained here for compatibility purposes.
1011     static void setFrameSystemTimestamp(std::shared_ptr<Frame> frame, uint64_t systemTimestamp) {
1012         // In order to compile, some high-version compilers will warn that the function parameters are not used.
1013         (void)frame;
1014         (void)systemTimestamp;
1015     }
1016
1017     static void setFrameDeviceTimestamp(std::shared_ptr<Frame> frame, uint64_t deviceTimestamp) {
1018         // In order to compile, some high-version compilers will warn that the function parameters are not used.
1019         (void)frame;
1020         (void)deviceTimestamp;
1021     }
1022 };
1023
1024 // Define the is() template function for the Frame class
1025 template <typename T> bool Frame::is() const {
1026     switch(this->getType0) {
1027         case OB_FRAME_IR_LEFT: // Follow
1028         case OB_FRAME_IR_RIGHT: // Follow
1029         case OB_FRAME_IR:
1030             return (typeid(T) == typeid(IRFrame)) || typeid(T) == typeid(VideoFrame));
1031         case OB_FRAME_DEPTH:
1032     }
1033 }

```

```
1058     return (typeid(T) == typeid(DepthFrame) || typeid(T) == typeid(VideoFrame));
1059 case OB_FRAME_COLOR:
1060     return (typeid(T) == typeid(ColorFrame) || typeid(T) == typeid(VideoFrame));
1061 case OB_FRAME_CONFIDENCE:
1062     return (typeid(T) == typeid(ConfidenceFrame) || typeid(T) == typeid(VideoFrame));
1063 case OB_FRAME_GYRO:
1064     return (typeid(T) == typeid(GyroFrame));
1065 case OB_FRAME_ACCEL:
1066     return (typeid(T) == typeid(AccelFrame));
1067 case OB_FRAME_POINTS:
1068     return (typeid(T) == typeid(PointsFrame));
1069 case OB_FRAME_SET:
1070     return (typeid(T) == typeid(FrameSet));
1071 default:
1072     std::cout << "ob::Frame::is() did not catch frame type: " << (int)this->getType() << std::endl;
1073     break;
1074 }
1075 return false;
1076 }
1077
1078 } // namespace ob
```

# MultipleDevices.h File Reference

This file contains the multiple devices related API which is used to control the synchronization between multiple devices and the synchronization between different sensor within single device. [More...](#)

```
#include "ObTypes.h"  
#include "Device.h"
```

[Go to the source code of this file.](#)

## Macros

```
#define ob_device_timer_reset ob_device_timestamp_reset
```

Alias for `ob_device_timestamp_reset` since it is more accurate.

## Functions

```
OB_EXPORT uint16_t ob_device_get_supported_multi_device_sync_mode_l  
(const ob_device *device, ob_error **error)
```

Get the supported multi device sync mode bitmap of the device.

```
OB_EXPORT void ob_device_set_multi_device_sync_config(ob_device  
*device, const ob_multi_device_sync_config *config,  
ob_error **error)
```

set the multi device sync configuration of the device.

```
OB_EXPORT ob_multi_device_sync_config ob_device_get_multi_device_sync_config(const ob_d  
*device, ob_error **error)
```

get the current multi device sync configuration of the de

```
OB_EXPORT void ob_device_trigger_capture (ob_device *device, ob_eri  
**error)
```

send the capture command to the device to trigger the capture.

```
OB_EXPORT void ob_device_set_timestamp_reset_config (ob_device *d  
const ob_device_timestamp_reset_config *config, ob_  
**error)
```

set the timestamp reset configuration of the device.

```
OB_EXPORT ob_device_timestamp_reset_config ob_device_get_timestamp_reset_config (ob_device *c  
ob_error **error)
```

get the timestamp reset configuration of the device.

```
OB_EXPORT void ob_device_timestamp_reset (ob_device *device, ob_e  
**error)
```

send the timestamp reset command to the device.

**OB\_EXPORT** void **ob\_device\_timer\_sync\_with\_host** (**ob\_device** \*device,  
**ob\_error** \*\*error)

synchronize the timer of the device with the host.

## Detailed Description

This file contains the multiple devices related API which is used to control the synchronization between multiple devices and the synchronization between different sensor within single device.

The synchronization between multiple devices is complex, and different models have different synchronization modes and limitations. please refer to the product manual for details.

As the Depth and Infrared are the same sensor physically, the behavior of the Infrared is same as the Depth in the synchronization mode.

Definition in file [MultipleDevices.h](#).

## Macro Definition Documentation

### ◆ **ob\_device\_timer\_reset**

```
#define ob_device_timer_reset ob_device_timestamp_reset
```

Alias for **ob\_device\_timestamp\_reset** since it is more accurate.

Definition at line [109](#) of file [MultipleDevices.h](#).

## Function Documentation

### ◆ **ob\_device\_get\_supported\_multi\_device\_sync\_mode\_bitmap()**

Get the supported multi device sync mode bitmap of the device.

For example, if the return value is 0b00001100, it means the device supports

**OB\_MULTI\_DEVICE\_SYNC\_MODE\_PRIMARY** and

**OB MULTI DEVICE SYNC MODE SECONDARY**: User can check the supported mode by the code:

```
if(supported_mode_bitmap & OB_MULTI_DEVICE_SYNC_MODE_FREE_RUN){  
    //support OB_MULTI_DEVICE_SYNC_MODE_FREE_RUN  
}  
if(supported_mode_bitmap & OB_MULTI_DEVICE_SYNC_MODE_STANDALONE){  
    //support OB_MULTI_DEVICE_SYNC_MODE_STANDALONE  
}  
// and so on
```

## Parameters

[in] **device** The device handle.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

`uint16_t` return the supported multi device sync mode bitmap of the device.

Referenced by [ob::Device::getSupportedMultiDeviceSyncModeBitmap\(\)](#).

- #### ◆ ob\_device\_set\_multi\_device\_sync\_config()

set the multi device sync configuration of the device.

## Parameters

[in] **device** The device handle.

[in] **config** The multi device sync configuration.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::setMultiDeviceSyncConfig\(\)](#).

- #### ◆ ob device get multi device sync config()

```
OB_EXPORT ob_multi_device_sync_config
ob_device_get_multi_device_sync_config
( const ob_device * device,
  ob_error **      error )
```

get the current multi device sync configuration of the device.

### Parameters

- [in] **device** The device handle.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_multi\_device\_sync\_config** return the multi device sync configuration of the device.

Referenced by **ob::Device::getMultiDeviceSyncConfig()**.

- ◆ **ob\_device\_trigger\_capture()**

```
OB_EXPORT void ob_device_trigger_capture ( ob_device * device,  
                                         ob_error ** error )
```

send the capture command to the device to trigger the capture.

The device will start one time capture after receiving the capture command when it is in the

### **OB\_MULTI\_DEVICE\_SYNC\_MODE\_SOFTWARE\_TRIGGERING**

#### **Attention**

The frequency of the user call this function multiplied by the number of frames per trigger should be less than the frame rate of the stream. The number of frames per trigger can be set by framesPerTrigger.

For some models, receive and execute the capture command will have a certain delay and performance consumption, so the frequency of calling this function should not be too high, please refer to the product manual for the specific supported frequency.

If the device is not in the

**OB\_MULTI\_DEVICE\_SYNC\_MODE\_HARDWARE\_TRIGGERING** mode, device will ignore the capture command.

#### **Parameters**

[in] **device** The device handle.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Device::triggerCapture()**.

#### ◆ **ob\_device\_set\_timestamp\_reset\_config()**

```
OB_EXPORT void ob_device_set_timestamp_reset_config ( ob_device *  
                                                       device,  
                                                       const ob_device_timestamp_reset_config * config,  
                                                       ob_error **  
                                                       error )
```

set the timestamp reset configuration of the device.

#### **Parameters**

[in] **device** The device handle.

[in] **config** The timestamp reset configuration.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Device::setTimestampResetConfig()**.

◆ [ob\\_device\\_get\\_timestamp\\_reset\\_config\(\)](#)

**OB\_EXPORT** [ob\\_device\\_timestamp\\_reset\\_config](#) ob\_device\_get\_timestamp\_reset\_config ( **ob\_device** \* device,  
**ob\_error** \*\* error )

get the timestamp reset configuration of the device.

**Parameters**

[in] **device** The device handle.

[out] **error** Pointer to an error object that will be set if an error occurs.

**Returns**

[ob\\_device\\_timestamp\\_reset\\_config](#) return the timestamp reset configuration of the device.

Referenced by [ob::Device::getTimestampResetConfig\(\)](#).

◆ [ob\\_device\\_timestamp\\_reset\(\)](#)

**OB\_EXPORT** void ob\_device\_timestamp\_reset ( **ob\_device** \* device,  
**ob\_error** \*\* error )

send the timestamp reset command to the device.

The device will reset the timer for calculating the timestamp for output frames to 0 after receiving the timestamp reset command when the timestamp reset function is enabled. The timestamp reset function can be enabled by call [ob\\_device\\_set\\_timestamp\\_reset\\_config](#).

**Attention**

If the stream of the device is started, the timestamp of the continuous frames output by the stream will jump once after the timestamp reset.

Due to the timer of device is not high-accuracy, the timestamp of the continuous frames output by the stream will drift after a long time. User can call this function periodically to reset the timer to avoid the timestamp drift, the recommended interval time is 60 minutes.

**Parameters**

[in] **device** The device handle.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::timestampReset\(\)](#).

◆ [ob\\_device\\_timer\\_sync\\_with\\_host\(\)](#)

```
OB_EXPORT void ob_device_timer_sync_with_host ( ob_device * device,  
                                              ob_error ** error )
```

synchronize the timer of the device with the host.

After calling this function, the timer of the device will be synchronized with the host. User can call this function to multiple devices to synchronize all timers of the devices.

### Attention

If the stream of the device is started, the timestamp of the continuous frames output by the stream will may jump once after the timer sync.

Due to the timer of device is not high-accuracy, the timestamp of the continuous frames output by the stream will drift after a long time. User can call this function periodically to synchronize the timer to avoid the timestamp drift, the recommended interval time is 60 minutes.

### Parameters

[in] **device** The device handle.

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Device::timerSyncWithHost\(\)](#).

# MultipleDevices.h

[Go to the documentation of this file.](#)

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
12
13 #pragma once
14
15 #ifndef __cplusplus
16 extern "C" {
17 #endif
18
19 #include "ObTypes.h"
20 #include "Device.h"
21
39 OB_EXPORT uint16_t ob_device_get_supported_multi_device_sync_mode_bitmap(const ob_device *de
    vice, ob_error **error);
40
48 OB_EXPORT void ob_device_set_multi_device_sync_config(ob_device *device, const ob_multi_device_
    sync_config *config, ob_error **error);
49
57 OB_EXPORT ob_multi_device_sync_config ob_device_get_multi_device_sync_config(const ob_device
    *device, ob_error **error);
58
72 OB_EXPORT void ob_device_trigger_capture(ob_device *device, ob_error **error);
73
81 OB_EXPORT void ob_device_set_timestamp_reset_config(ob_device *device, const ob_device_timestamp_
    p_reset_config *config, ob_error **error);
82
90 OB_EXPORT ob_device_timestamp_reset_config ob_device_get_timestamp_reset_config(ob_device *de
    vice, ob_error **error);
91
104 OB_EXPORT void ob_device_timestamp_reset(ob_device *device, ob_error **error);
105
109 #define ob_device_timer_reset ob_device_timestamp_reset
110
123 OB_EXPORT void ob_device_timer_sync_with_host(ob_device *device, ob_error **error);
124
125 #ifndef __cplusplus
126 } // extern "C"
127 #endif
128
```

# ObSensor.h File Reference

This file serves as the C entrance for the OrbbecSDK library. It includes all necessary header files for OrbbecSDK usage. [More...](#)

```
#include <libobssensor/h/Context.h>
#include <libobssensor/h/Device.h>
#include <libobssensor/h/Error.h>
#include <libobssensor/h/Filter.h>
#include <libobssensor/h/Frame.h>
#include <libobssensor/h/ObTypes.h>
#include <libobssensor/h/Pipeline.h>
#include <libobssensor/h/Property.h>
#include <libobssensor/h/Sensor.h>
#include <libobssensor/h/StreamProfile.h>
#include <libobssensor/h/Version.h>
#include <libobssensor/h/TypeHelper.h>
#include <libobssensor/h/RecordPlayback.h>
```

[Go to the source code of this file.](#)

## Detailed Description

This file serves as the C entrance for the OrbbecSDK library. It includes all necessary header files for OrbbecSDK usage.

Definition in file [ObSensor.h](#).

# ObSensor.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.  
2 // Licensed under the MIT License.  
3  
9 #pragma once  
10  
11 #include <libobssensor/h/Context.h>  
12 #include <libobssensor/h/Device.h>  
13 #include <libobssensor/h/Error.h>  
14 #include <libobssensor/h/Filter.h>  
15 #include <libobssensor/h/Frame.h>  
16 #include <libobssensor/h/ObTypes.h>  
17 #include <libobssensor/h/Pipeline.h>  
18 #include <libobssensor/h/Property.h>  
19 #include <libobssensor/h/Sensor.h>  
20 #include <libobssensor/h/StreamProfile.h>  
21 #include <libobssensor/h/Version.h>  
22 #include <libobssensor/h/TypeHelper.h>  
23 #include <libobssensor/h/RecordPlayback.h>
```

Generated on for OrbbecSDK by  1.14.0

# ObSensor.hpp File Reference

This is the main entry point for the OrbbecSDK C++ library. It includes all necessary header files for using the library. [More...](#)

```
#include <libobssensor.hpp/Context.hpp>
#include <libobssensor.hpp/Device.hpp>
#include <libobssensor.hpp/Error.hpp>
#include <libobssensor.hpp/Filter.hpp>
#include <libobssensor.hpp/Frame.hpp>
#include <libobssensor.hpp/Pipeline.hpp>
#include <libobssensor.hpp/RecordPlayback.hpp>
#include <libobssensor.hpp/Sensor.hpp>
#include <libobssensor.hpp/StreamProfile.hpp>
#include <libobssensor.hpp/Version.hpp>
#include <libobssensor.hpp/TypeHelper.hpp>
```

[Go to the source code of this file.](#)

## Detailed Description

This is the main entry point for the OrbbecSDK C++ library. It includes all necessary header files for using the library.

Definition in file [ObSensor.hpp](#).

# ObSensor.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.  
2 // Licensed under the MIT License.  
3  
9 #pragma once  
10  
11 #include <libobssensor.hpp/Context.hpp>  
12 #include <libobssensor.hpp/Device.hpp>  
13 #include <libobssensor.hpp/Error.hpp>  
14 #include <libobssensor.hpp/Filter.hpp>  
15 #include <libobssensor.hpp/Frame.hpp>  
16 #include <libobssensor.hpp/Pipeline.hpp>  
17 #include <libobssensor.hpp/RecordPlayback.hpp>  
18 #include <libobssensor.hpp/Sensor.hpp>  
19 #include <libobssensor.hpp/StreamProfile.hpp>  
20 #include <libobssensor.hpp/Version.hpp>  
21 #include <libobssensor.hpp/TypeHelper.hpp>
```

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ObTypes.h File Reference

Provide structs commonly used in the SDK, enumerating constant definitions. [More...](#)

```
#include "Export.h"
#include <stdbool.h>
#include <stdint.h>
```

[Go to the source code of this file.](#)

## Classes

### struct **ob\_error**

The error class exposed by the SDK, users can get detailed error information according to the error. [More...](#)

### struct **OBDDataChunk**

Structure for transmitting data blocks. [More...](#)

### struct **OBIntPropertyRange**

Structure for integer range. [More...](#)

### struct **OBFloatPropertyRange**

Structure for float range. [More...](#)

### struct **OBUInt16PropertyRange**

Structure for float range. [More...](#)

### struct **OBUInt8PropertyRange**

Structure for float range. [More...](#)

### struct **OBBoolPropertyRange**

Structure for boolean range. [More...](#)

### struct **OBCameraIntrinsic**

Structure for camera intrinsic parameters. [More...](#)

### struct **OBAccelIntrinsic**

Structure for accelerometer intrinsic parameters. [More...](#)

### struct **OBGyroIntrinsic**

Structure for gyroscope intrinsic parameters. [More...](#)

### struct **OBCameraDistortion**

Structure for distortion parameters. [More...](#)

### struct **OBD2CTransform**

Structure for rotation/transformation. [More...](#)

### struct **OBCameraParam**

Structure for camera parameters. [More...](#)

struct **OBPresetResolutionConfig**

struct **OBCalibrationParam**

calibration parameters [More...](#)

struct **ob\_margin\_filter\_config**

Configuration for depth margin filter. [More...](#)

struct **OBMGCFilterConfig**

Configuration for mgc filter. [More...](#)

struct **OBRect**

Rectangle. [More...](#)

struct **OBAccelValue**

Data structures for accelerometers and gyroscopes. [More...](#)

struct **OBDeviceTemperature**

Temperature parameters of the device (unit: Celsius) [More...](#)

struct **OBDisparityParam**

disparity parameters for disparity based camera [More...](#)

struct **OBPoint**

3D point structure in the SDK [More...](#)

struct **OBPoint2f**

2D point structure in the SDK [More...](#)

struct **OBXYTables**

struct **OBCColorPoint**

3D point structure with color information [More...](#)

struct **OBCompressionParams**

struct **OBTofExposureThresholdControl**

TOF Exposure Threshold. [More...](#)

struct **OBDeviceSyncConfig**

Device synchronization configuration. [More...](#)

struct **OBDepthWorkMode**

Depth work mode. [More...](#)

struct **OBSequenceIdItem**

SequenceId filter list item. [More...](#)

struct **OBSpatialAdvancedFilterParams**

struct **OBEdgeNoiseRemovalFilterParams**

struct **OBNoiseRemovalFilterParams**

struct **OBProtocolVersion**

Control command protocol version number. [More...](#)

struct **OBNetIpConfig**  
IP address configuration for network devices (IPv4) [More...](#)

struct **ob\_multi\_device\_sync\_config**  
The synchronization configuration of the device. [More...](#)

struct **ob\_device\_timestamp\_reset\_config**  
The timestamp reset configuration of the device. [More...](#)

struct **BASELINE\_CALIBRATION\_PARAM**  
Baseline calibration parameters. [More...](#)

struct **HDR\_CONFIG**  
HDR Configuration. [More...](#)

struct **AE\_ROI**  
The rect of the region of interest. [More...](#)

struct **OBFilterConfigSchemaItem**  
Configuration Item for the filter. [More...](#)

struct **OBDeviceSerialNumber**  
struct of serial number [More...](#)

struct **OBDispOffsetConfig**  
Disparity offset interleaving configuration. [More...](#)

## Macros

```
#define OB_WIDTH_ANY 0
#define OB_HEIGHT_ANY 0
#define OB_FPS_ANY 0
#define OB_FORMAT_ANY OB_FORMAT_UNKNOWN
#define OB_PROFILE_DEFAULT 0
#define OB_DEFAULT_STRIDE_BYTES 0
#define OB_ACCEL_FULL_SCALE_RANGE_ANY OB_ACCEL_FS_UNKNOWN
#define OB_ACCEL_SAMPLE_RATE_ANY OB_SAMPLE_RATE_UNKNOWN
#define OB_GYRO_FULL_SCALE_RANGE_ANY OB_GYRO_FS_UNKNOWN
#define OB_GYRO_SAMPLE_RATE_ANY OB_SAMPLE_RATE_UNKNOWN
#define OB_PATH_MAX (1024)
    maximum path length
#define OB_LOG_SEVERITY_NONE OB_LOG_SEVERITY_OFF
#define OB_FORMAT_RGB888 OB_FORMAT_RGB
#define OB_FORMAT_MJPEG OB_FORMAT_MJPG
#define IS_FIXED_SIZE_FORMAT(format)
#define IS_PACKED_FORMAT(format)
```

```

#define FORMAT_MJPEG_TO_I420 FORMAT_MJPG_TO_I420
#define FORMAT_MJPEG_TO_NV21 FORMAT_MJPG_TO_NV21
#define FORMAT_MJPEG_TO_BGRA FORMAT_MJPG_TO_BGRA
#define FORMAT_YUYV_TO_RGB888 FORMAT_YUYV_TO_RGB
#define FORMAT_I420_TO_RGB888 FORMAT_I420_TO_RGB
#define FORMAT_NV21_TO_RGB888 FORMAT_NV21_TO_RGB
#define FORMAT_NV12_TO_RGB888 FORMAT_NV12_TO_RGB
#define FORMAT_UYVY_TO_RGB888 FORMAT_UYVY_TO_RGB
#define FORMAT_MJPG_TO_RGB888 FORMAT_MJPG_TO_RGB
#define FORMAT_MJPG_TO_BGR888 FORMAT_MJPG_TO_BGR
#define FORMAT_MJPEG_TO_RGB888 FORMAT_MJPEG_TO_RGB
#define FORMAT_MJPEG_TO_BGR888 FORMAT_MJPEG_TO_BGR
#define FORMAT_RGB888_TO_BGR FORMAT_RGB_TO_BGR
#define OBDeviceIpAddrConfig OBNetIpConfig
#define ob_device_ip_addr_config OBNetIpConfig
#define OB_FRAME_AGGREGATE_OUTPUT_FULL_FRAME_REQUIRE OB_FRAME_AGGREGATE_
#define OB_FRAME_METADATA_TYPE_LASER_POWER_MODE OB_FRAME_METADATA_TYPE_I
#define OB_FRAME_METADATA_TYPE_EMITTER_MODE OB_FRAME_METADATA_TYPE_LASER_
#define ob_filter_callback ob_frame_callback
#define ob_playback_callback ob_frame_callback
#define ob_is_video_sensor_type(sensor_type)
    Check if the sensor_type is a video sensor.
#define ob_is_video_stream_type(stream_type)
    check if the stream_type is a video stream
#define is_ir_sensor(sensor_type)
    Check if sensor_type is an IR sensor.
#define isIRSensor is_ir_sensor
#define is_ir_stream(stream_type)
    Check if stream_type is an IR stream.
#define isIRStream is_ir_stream
#define is_ir_frame(frame_type)
    Check if frame_type is an IR frame.
#define isIRFrame is_ir_frame
#define OB_DEFAULT_DECRYPT_KEY (nullptr)
    The default Decrypt Key.

```

## Typedefs

```
typedef struct ob_context_t ob_context
    typedef struct ob_device_t ob_device
    typedef struct ob_device_info_t ob_device_info
    typedef struct ob_device_list_t ob_device_list
    typedef struct ob_record_device_t ob_record_device
    typedef struct ob_playback_device_t ob_playback_device
    typedef struct ob_camera_param_list_t ob_camera_param_list
        typedef struct ob_sensor_t ob_sensor
            typedef struct ob_sensor_list_t ob_sensor_list
            typedef struct ob_stream_profile_t ob_stream_profile
            typedef struct ob_stream_profile_list_t ob_stream_profile_list
                typedef struct ob_frame_t ob_frame
                typedef struct ob_filter_t ob_filter
                typedef struct ob_filter_list_t ob_filter_list
                typedef struct ob_pipeline_t ob_pipeline
                typedef struct ob_config_t ob_config
            typedef struct ob_depth_work_mode_list_t ob_depth_work_mode_list
                typedef struct ob_device_preset_list_t ob_device_preset_list
            typedef struct ob_filter_config_schema_list_t ob_filter_config_schema_list
            typedef struct ob_device_frame_interleave_list_t ob_device_frame_interleave_list
            typedef struct ob_preset_resolution_config_list_t ob_preset_resolution_config_list
            typedef enum OBPermissionType ob_permission_type
                typedef enum OBStatus ob_status
                typedef enum OBLogSeverity ob_log_severity
                typedef enum OBLogSeverity DEVICE_LOG_SEVERITY_LEVEL
                typedef enum OBLogSeverity OBDeviceLogSeverityLevel
                typedef enum OBLogSeverity ob_device_log_severity_level
            typedef enum OBExceptionType ob_exception_type
                typedef struct ob_error ob_error
                    The error class exposed by the SDK, users
                    can get detailed error information according to
                    the error.
                typedef enum OBSensorType ob_sensor_type
                typedef enum OBStreamType ob_stream_type
                typedef enum OBFrameType ob_frame_type
                typedef enum OBPixelType ob_pixel_type
                typedef enum OBFormat ob_format
```

```
typedef enum OBUpgradeState OBFwUpdateState
typedef enum OBUpgradeState ob_upgrade_state
typedef enum OBUpgradeState ob_fw_update_state
typedef enum OBFileTranState ob_file_tran_state
typedef enum OBDataTranState ob_data_tran_state
typedef struct OBDataChunk ob_data_chunk
typedef struct OBIntPropertyRange ob_int_property_range
typedef struct OBFloatPropertyRange ob_float_property_range
typedef struct OBUInt16PropertyRange ob_uint16_property_range
typedef struct OB UInt8PropertyRange ob_uint8_property_range
typedef struct OBBoolPropertyRange ob_bool_property_range
typedef enum OBCameraDistortionModel ob_camera_distortion_model
typedef struct OBCameraIntrinsic ob_camera_intrinsic
typedef struct OBAccelIntrinsic ob_accel_intrinsic
typedef struct OBGyroIntrinsic ob_gyro_intrinsic
typedef struct OBCameraDistortion ob_camera_distortion
typedef struct OBD2CTransform ob_d2c_transform
typedef struct OBD2CTransform OBTransform
typedef struct OBD2CTransform ob_transform
typedef struct OBD2CTransform OBExtrinsic
typedef struct OBD2CTransform ob_extrinsic
typedef struct OBCameraParam ob_camera_param
typedef struct OBPresetResolutionConfig ob_preset_resolution_ratio_config
typedef struct OBCalibrationParam ob_calibration_param
typedef struct ob_margin_filter_config OBMarginFilterConfig
typedef struct OBMGCFilterConfig ob_mgc_filter_config
typedef enum OBAccelMode ob_align_mode
typedef enum OBCameraPerformanceMode ob_camera_performance_mode
typedef struct OBRect ob_rect
typedef enum OBConvertFormat ob_convert_format
typedef enum OBIMUSampleRate OBGyroSampleRate
typedef enum OBIMUSampleRate ob_gyro_sample_rate
typedef enum OBIMUSampleRate OBAccelSampleRate
typedef enum OBIMUSampleRate ob_accel_sample_rate
typedef enum OBIMUSampleRate OB_SAMPLE_RATE
typedef enum OBGyroFullScaleRange ob_gyro_full_scale_range
typedef enum OBGyroFullScaleRange OB_GYRO_FULL_SCALE_RANGE
```

```
typedef enum OBAccelFullScaleRange ob_accel_full_scale_range
typedef enum OBAccelFullScaleRange OB_ACCEL_FULL_SCALE_RANGE
    typedef struct OBAccelValue OBGyroValue
        typedef struct OBAccelValue OBFloat3D
            typedef struct OBAccelValue ob_accel_value
            typedef struct OBAccelValue ob_gyro_value
            typedef struct OBAccelValue ob_float_3d
                typedef uint64_t OBDeviceState
                    Device state.

                    typedef uint64_t ob_device_state

typedef struct OBDeviceTemperature ob_device_temperature
typedef struct OBDeviceTemperature DEVICE_TEMPERATURE
typedef enum OBDepthCroppingMode ob_depth_cropping_mode
typedef enum OBDepthCroppingMode OB_DEPTH_CROPPING_MODE
    typedef enum OBDeviceType ob_device_type
    typedef enum OBDeviceType OB_DEVICE_TYPE
    typedef enum OBMediaType ob_media_type
    typedef enum OBMediaType OB_MEDIA_TYPE
    typedef enum OBMediaState ob_media_state
    typedef enum OBMediaState OB_MEDIA_STATE_EM
typedef enum OBDepthPrecisionLevel ob_depth_precision_level
typedef enum OBDepthPrecisionLevel OB_DEPTH_PRECISION_LEVEL
typedef enum OBDepthPrecisionLevel OBDepthUnit
typedef enum OBDepthPrecisionLevel ob_depth_unit
    typedef struct OBDisparityParam ob_disparity_param
    typedef enum OBTofFilterRange ob_tof_filter_range
    typedef enum OBTofFilterRange TOF_FILTER_RANGE
        typedef struct OBPoint ob_point
        typedef struct OBPoint OBPoint3f
        typedef struct OBPoint ob_point3f
        typedef struct OBPoint2f ob_point2f
    typedef struct OBXYTables ob_xy_tables
    typedef struct OBColorPoint ob_color_point
typedef enum OBCompressionMode ob_compression_mode
typedef enum OBCompressionMode OB_COMPRESSION_MODE
typedef struct OBCompressionParams ob_compression_params
typedef struct OBCompressionParams OB_COMPRESSION_PARAMS
```

```
typedef struct OBTofExposureThresholdControl ob_tof_exposure_threshold_control
typedef struct OBTofExposureThresholdControl TOF_EXPOSURE_THRESHOLD_CONTROL

    typedef enum OBSyncMode ob_sync_mode
    typedef enum OBSyncMode OB_SYNC_MODE
    typedef struct OBDeviceSyncConfig ob_device_sync_config
    typedef struct OBDeviceSyncConfig OB_DEVICE_SYNC_CONFIG
    typedef enum OBDepthWorkModeTag ob_depth_work_mode_tag
    typedef struct OBDepthWorkMode ob_depth_work_mode
    typedef struct OBSequenceIdItem ob_sequence_id_item
    typedef enum OBHoleFillingMode ob_hole_filling_mode

    typedef struct OBSpatialAdvancedFilterParams ob_spatial_advanced_filter_params
typedef enum OB_EDGE_NOISE_REMOVAL_TYPE OBEdgeNoiseRemovalType
typedef enum OB_EDGE_NOISE_REMOVAL_TYPE ob_edge_noise_removal_type
    typedef struct OBEdgeNoiseRemovalFilterParams ob_edge_noise_removal_filter_params
    typedef enum OB_DDO_NOISE_REMOVAL_TYPE OBDDONoiseRemovalType
                                Denoising method.

    typedef enum OB_DDO_NOISE_REMOVAL_TYPE ob_ddo_noise_removal_type
    typedef struct OBNoiseRemovalFilterParams ob_noise_removal_filter_params
        typedef struct OBProtocolVersion ob_protocol_version
        typedef enum OB_CMD_VERSION OBCmdVersion
        typedef enum OB_CMD_VERSION ob_cmd_version
            typedef struct OBNetIpConfig ob_net_ip_config
            typedef struct OBNetIpConfig DEVICE_IP_ADDR_CONFIG
            typedef enum OBCommunicationType ob_communication_type
            typedef enum OBCommunicationType OB_COMMUNICATION_TYPE
                typedef enum OBUSBPowerState ob_usb_power_state
                typedef enum OBDCPowerState ob_dc_power_state
                typedef enum ob_rotate_degree_type OBRotateDegreeType
                typedef enum ob_power_line_freq_mode OBPowerLineFreqMode

typedef enum OB_FRAME_AGGREGATE_OUTPUT_MODE OBFrameAggregateOutputMode
typedef enum OB_FRAME_AGGREGATE_OUTPUT_MODE ob_frame_aggregate_output_mode

    typedef enum OB_COORDINATE_SYSTEM_TYPE OBCoordinateSystemType
    typedef enum OB_COORDINATE_SYSTEM_TYPE ob_coordinate_system_type
typedef enum OB_DEVICE_DEVELOPMENT_MODE OBDeviceDevelopmentMode
typedef enum OB_DEVICE_DEVELOPMENT_MODE ob_device_development_mode

    typedef enum ob_multi_device_sync_mode OBMultiDeviceSyncMode
    typedef struct ob_multi_device_sync_config OBMultiDeviceSyncConfig
```



```

typedef void(* ob_device_changed_callback) (ob_device_list
                                         *removed, ob_device_list *added, void
                                         *user_data)
                                         Callback for device change.

typedef void(* ob_frame_callback) (ob_frame *frame, void
                                         *user_data)
                                         Callback for frame.

typedef void(* ob_frameset_callback) (ob_frame *frameset,
                                         void *user_data)
                                         Callback for frameset.

typedef void ob_frame_destroy_callback(uint8_t *buffer,
                                         void *user_data)
                                         Customize the delete callback.

typedef void ob_log_callback(ob_log_severity severity,
                           const char *message, void *user_data)
                           Callback for receiving log.

typedef void(* ob_playback_status_changed_callback)
            (ob_playback_status status, void *user_data)

```

## Enumerations

```

enum OBPermissionType {
    OB_PERMISSION_DENY = 0 , OB_PERMISSION_READ = 1 , OB_PERMISSION_WRITE
    = 2 , OB_PERMISSION_READ_WRITE = 3 ,
    OB_PERMISSION_ANY = 255
}

```

the permission type of api or property [More...](#)

```

enum OBStatus { OB_STATUS_OK= 0 , OB_STATUS_ERROR = 1 }

```

error code [More...](#)

```

enum OBLogSeverity {
    OB_LOG_SEVERITY_DEBUG , OB_LOG_SEVERITY_INFO ,
    OB_LOG_SEVERITY_WARN , OB_LOG_SEVERITY_ERROR ,
    OB_LOG_SEVERITY_FATAL , OB_LOG_SEVERITY_OFF
}

```

log level, the higher the level, the stronger the log filter [More...](#)

```
enum OBExceptionType {
    OB_EXCEPTION_TYPE_UNKNOWN, OB_EXCEPTION_STD_EXCEPTION,
    OB_EXCEPTION_TYPE_CAMERA_DISCONNECTED,
    OB_EXCEPTION_TYPE_PLATFORM,
    OB_EXCEPTION_TYPE_INVALID_VALUE,
    OB_EXCEPTION_TYPE_WRONG_API_CALL_SEQUENCE,
    OB_EXCEPTION_TYPE_NOT_IMPLEMENTED, OB_EXCEPTION_TYPE_IO,
    OB_EXCEPTION_TYPE_MEMORY,
    OB_EXCEPTION_TYPE_UNSUPPORTED_OPERATION
}
```

The exception types in the SDK, through the exception type, you can easily determine the specific type of error. For detailed error API interface functions and error logs, please refer to the information of [ob\\_error](#). [More...](#)

```
enum OBSensorType {
    OB_SENSOR_UNKNOWN=0, OB_SENSOR_IR=1, OB_SENSOR_COLOR=2,
    OB_SENSOR_DEPTH=3,
    OB_SENSOR_ACCEL=4, OB_SENSOR_GYRO=5, OB_SENSOR_IR_LEFT=6,
    OB_SENSOR_IR_RIGHT=7,
    OB_SENSOR_RAW_PHASE=8, OB_SENSOR_CONFIDENCE=9,
    OB_SENSOR_TYPE_COUNT
}
```

Enumeration value describing the sensor type. [More...](#)

```
enum OBStreamType {
    OB_STREAM_UNKNOWN=-1, OB_STREAM_VIDEO=0, OB_STREAM_IR=1,
    OB_STREAM_COLOR=2,
    OB_STREAM_DEPTH=3, OB_STREAM_ACCEL=4, OB_STREAM_GYRO=5,
    OB_STREAM_IR_LEFT=6,
    OB_STREAM_IR_RIGHT=7, OB_STREAM_RAW_PHASE=8,
    OB_STREAM_CONFIDENCE=9, OB_STREAM_TYPE_COUNT
}
```

Enumeration value describing the type of data stream. [More...](#)

```
enum OBFrameType {
    OB_FRAME_UNKNOWN=-1, OB_FRAME_VIDEO=0, OB_FRAME_IR=1,
    OB_FRAME_COLOR=2,
    OB_FRAME_DEPTH=3, OB_FRAME_ACCEL=4, OB_FRAME_SET=5,
    OB_FRAME_POINTS=6,
    OB_FRAME_GYRO=7, OB_FRAME_IR_LEFT=8, OB_FRAME_IR_RIGHT=9,
    OB_FRAME_RAW_PHASE=10,
    OB_FRAME_CONFIDENCE=11, OB_FRAME_TYPE_COUNT
}
```

Enumeration value describing the type of frame. [More...](#)

```

enum OBPixelType {
    OB_PIXEL_UNKNOWN= -1 , OB_PIXEL_DEPTH= 0 , OB_PIXEL_DISPARITY= 2 ,
    OB_PIXEL_RAW_PHASE= 3 ,
    OB_PIXEL_TOF_DEPTH= 4
}
Enumeration value describing the pixel type of frame (usually used for depth frame) More...

enum OBFormat {
    OB_FORMAT_UNKNOWN= -1 , OB_FORMAT_YUYV= 0 , OB_FORMAT_YUY2= 1 ,
    OB_FORMAT_UYVY= 2 ,
    OB_FORMAT_NV12= 3 , OB_FORMAT_NV21= 4 , OB_FORMAT_MJPG= 5 ,
    OB_FORMAT_H264= 6 ,
    OB_FORMAT_H265= 7 , OB_FORMAT_Y16= 8 , OB_FORMAT_Y8= 9 ,
    OB_FORMAT_Y10= 10 ,
    OB_FORMAT_Y11= 11 , OB_FORMAT_Y12= 12 , OB_FORMAT_GRAY= 13 ,
    OB_FORMAT_HEVC= 14 ,
    OB_FORMAT_I420= 15 , OB_FORMAT_ACCEL= 16 , OB_FORMAT_GYRO= 17 ,
    OB_FORMAT_POINT= 19 ,
    OB_FORMAT_RGB_POINT= 20 , OB_FORMAT_RLE= 21 , OB_FORMAT_RGB= 22 ,
    OB_FORMAT_BGR= 23 ,
    OB_FORMAT_Y14= 24 , OB_FORMAT_BGRA= 25 , OB_FORMAT_COMPRESSED= 26
    , OB_FORMAT_RVL= 27 ,
    OB_FORMAT_Z16= 28 , OB_FORMAT_YV12= 29 , OB_FORMAT_BA81= 30 ,
    OB_FORMAT_RGBA= 31 ,
    OB_FORMAT_BYR2= 32 , OB_FORMAT_RW16= 33 , OB_FORMAT_Y12C4= 34
}
Enumeration value describing the pixel format. More...

enum OBUpgradeState {
    STAT_DONE_WITH_DUPLICATES= 6 , STAT_VERIFY_SUCCESS= 5 ,
    STAT_FILE_TRANSFER= 4 , STAT_DONE= 3 ,
    STAT_IN_PROGRESS= 2 , STAT_START= 1 , STAT_VERIFY_IMAGE= 0 ,
    ERR_VERIFY= -1 ,
    ERR_PROGRAM= -2 , ERR_ERASE= -3 , ERR_FLASH_TYPE= -4 , ERR_IMAGE_SIZE=
    -5 ,
    ERR_OTHER= -6 , ERR_DDR= -7 , ERR_TIMEOUT= -8 , ERR_MISMATCH= -9 ,
    ERR_UNSUPPOT_DEV= -10 , ERR_INVALID_COUNT= -11
}
Enumeration value describing the firmware upgrade status. More...

```

```
enum OBFileTranState {  
    FILE_TRAN_STAT_TRANSFER = 2 , FILE_TRAN_STAT_DONE = 1 ,  
    FILE_TRAN_STAT_PREPAR = 0 , FILE_TRAN_ERR_DDR = -1 ,  
    FILE_TRAN_ERR_NOT_ENOUGH_SPACE = -2 ,  
    FILE_TRAN_ERR_PATH_NOT_WRITABLE = -3 , FILE_TRAN_ERR_MD5_ERROR = -4 ,  
    FILE_TRAN_ERR_WRITE_FLASH_ERROR = -5 ,  
    FILE_TRAN_ERR_TIMEOUT = -6  
}
```

Enumeration value describing the file transfer status. [More...](#)

```
enum OBDATATranState {  
    DATA_TRAN_STAT_VERIFY_DONE = 4 , DATA_TRAN_STAT_STOPPED = 3 ,  
    DATA_TRAN_STAT_DONE = 2 , DATA_TRAN_STAT VERIFYING = 1 ,  
    DATA_TRAN_STAT_TRANSFERRING = 0 , DATA_TRAN_ERR_BUSY = -1 ,  
    DATA_TRAN_ERR_UNSUPPORTED = -2 , DATA_TRAN_ERR_TRAN_FAILED = -3 ,  
    DATA_TRAN_ERR_VERIFY_FAILED = -4 , DATA_TRAN_ERR_OTHER = -5  
}
```

Enumeration value describing the data transfer status. [More...](#)

```
enum OBCameraDistortionModel {  
    OB_DISTORTION_NONE , OB_DISTORTION_MODIFIED_BROWN_CONRADY ,  
    OB_DISTORTION_INVERSE_BROWN_CONRADY ,  
    OB_DISTORTION_BROWN_CONRADY ,  
    OB_DISTORTION_BROWN_CONRADY_K6 , OB_DISTORTION_KANNALA_BRANDT4  
}
```

Distortion model: defines how pixel coordinates should be mapped to sensor coordinates. [More...](#)

```
enum OBAlignMode { ALIGN_DISABLE , ALIGN_D2C_HW_MODE , ALIGN_D2C_SW_MODE }  
Alignment mode. More...
```

```
enum OBCameraPerformanceMode { ADAPTIVE_PERFORMANCE_MODE ,  
    HIGH_PERFORMANCE_MODE }  
Camera performance mode. More...
```

```
enum OBConvertFormat {  
    FORMAT_YUYV_TO_RGB = 0 , FORMAT_I420_TO_RGB , FORMAT_NV21_TO_RGB ,  
    FORMAT_NV12_TO_RGB ,  
    FORMAT_MJPG_TO_I420 , FORMAT_RGB_TO_BGR , FORMAT_MJPG_TO_NV21 ,  
    FORMAT_MJPG_TO_RGB ,  
    FORMAT_MJPG_TO_BGR , FORMAT_MJPG_TO_BGRA , FORMAT_UYVY_TO_RGB ,  
    FORMAT_BGR_TO_RGB ,  
    FORMAT_MJPG_TO_NV12 , FORMAT_YUYV_TO_BGR , FORMAT_YUYV_TO_RGBA ,  
    FORMAT_YUYV_TO_BGRA ,  
    FORMAT_YUV_TO_Y16 , FORMAT_YUYV_TO_Y8 , FORMAT_RGBA_TO_RGB ,  
    FORMAT_BGRA_TO_BGR ,  
    FORMAT_Y16_TO_RGB , FORMAT_Y8_TO_RGB  
}
```

Enumeration of format conversion types. [More...](#)

```
enum OBIMUSampleRate {
    OB_SAMPLE_RATE_UNKNOWN= 0 , OB_SAMPLE_RATE_1_5625_HZ= 1 ,
    OB_SAMPLE_RATE_3_125_HZ= 2 , OB_SAMPLE_RATE_6_25_HZ= 3 ,
    OB_SAMPLE_RATE_12_5_HZ= 4 , OB_SAMPLE_RATE_25_HZ= 5 ,
    OB_SAMPLE_RATE_50_HZ= 6 , OB_SAMPLE_RATE_100_HZ= 7 ,
    OB_SAMPLE_RATE_200_HZ= 8 , OB_SAMPLE_RATE_500_HZ= 9 ,
    OB_SAMPLE_RATE_1_KHZ= 10 , OB_SAMPLE_RATE_2_KHZ= 11 ,
    OB_SAMPLE_RATE_4_KHZ= 12 , OB_SAMPLE_RATE_8_KHZ= 13 ,
    OB_SAMPLE_RATE_16_KHZ= 14 , OB_SAMPLE_RATE_32_KHZ= 15 ,
    OB_SAMPLE_RATE_400_HZ= 16 , OB_SAMPLE_RATE_800_HZ= 17
}
```

Enumeration of IMU sample rate values (gyroscope or accelerometer) [More...](#)

```
enum OBGyroFullScaleRange {
    OB_GYRO_FS_UNKNOWN= -1 , OB_GYRO_FS_16dps= 1 , OB_GYRO_FS_31dps= 2 ,
    OB_GYRO_FS_62dps= 3 ,
    OB_GYRO_FS_125dps= 4 , OB_GYRO_FS_250dps= 5 , OB_GYRO_FS_500dps= 6 ,
    OB_GYRO_FS_1000dps= 7 ,
    OB_GYRO_FS_2000dps= 8 , OB_GYRO_FS_400dps= 9 , OB_GYRO_FS_800dps= 10
}
```

Enumeration of gyroscope ranges. [More...](#)

```
enum OBAccelFullScaleRange {
    OB_ACCEL_FS_UNKNOWN= -1 , OB_ACCEL_FS_2g= 1 , OB_ACCEL_FS_4g= 2 ,
    OB_ACCEL_FS_8g= 3 ,
    OB_ACCEL_FS_16g= 4 , OB_ACCEL_FS_3g= 5 , OB_ACCEL_FS_6g= 6 ,
    OB_ACCEL_FS_12g= 7 ,
    OB_ACCEL_FS_24g= 8
}
```

Enumeration of accelerometer ranges. [More...](#)

```
enum OBDepthCroppingMode { DEPTH_CROPPING_MODE_AUTO= 0 ,
    DEPTH_CROPPING_MODE_CLOSE= 1 , DEPTH_CROPPING_MODE_OPEN= 2 }
```

Enumeration for depth crop modes. [More...](#)

```
enum OBDeviceType { OB_DEVICE_TYPE_UNKNOWN= -1 ,
    OB_STRUCTURED_LIGHT_MONOCULAR_CAMERA= 0 ,
    OB_STRUCTURED_LIGHT_BINOCULAR_CAMERA= 1 , OB_TOF_CAMERA= 2 }
```

Enumeration for device types. [More...](#)

```

enum OBMediaType {
    OB_MEDIA_COLOR_STREAM=1, OB_MEDIA_DEPTH_STREAM=2,
    OB_MEDIA_IR_STREAM=4, OB_MEDIA_GYRO_STREAM=8,
    OB_MEDIA_ACCEL_STREAM=16, OB_MEDIA_CAMERA_PARAM=32,
    OB_MEDIA_DEVICE_INFO=64, OB_MEDIA_STREAM_INFO=128,
    OB_MEDIA_IR_LEFT_STREAM=256, OB_MEDIA_IR_RIGHT_STREAM=512,
    OB_MEDIA_ALL
}

```

Enumeration for types of media to record or playback. [More...](#)

```

enum OBMediaState { OB_MEDIA_BEGIN=0, OB_MEDIA_PAUSE, OB_MEDIA_RESUME,
OB_MEDIA_END }

```

Enumeration for record playback status. [More...](#)

```

enum OBDepthPrecisionLevel {
    OB_PRECISION_1MM, OB_PRECISION_0MM8, OB_PRECISION_0MM4,
    OB_PRECISION_0MM1,
    OB_PRECISION_0MM2, OB_PRECISION_0MM5, OB_PRECISION_0MM05,
    OB_PRECISION_UNKNOWN,
    OB_PRECISION_COUNT
}

```

Enumeration for depth precision levels. [More...](#)

```

enum OBTofFilterRange { OB_TOF_FILTER_RANGE_CLOSE=0,
OB_TOF_FILTER_RANGE_MIDDLE=1, OB_TOF_FILTER_RANGE_LONG=2,
OB_TOF_FILTER_RANGE_DEBUG=100 }

```

Enumeration for TOF filter scene ranges. [More...](#)

```

enum OBCompressionMode { OB_COMPRESSION_LOSSLESS=0,
OB_COMPRESSION_LOSSY=1 }

```

Compression mode. [More...](#)

```

enum OBSyncMode {
    OB_SYNC_MODE_CLOSE=0x00, OB_SYNC_MODE_STANDALONE=0x01,
    OB_SYNC_MODE_PRIMARY=0x02, OB_SYNC_MODE_SECONDARY=0x03,
    OB_SYNC_MODE_PRIMARY MCU_TRIGGER=0x04,
    OB_SYNC_MODE_PRIMARY IR_TRIGGER=0x05,
    OB_SYNC_MODE_PRIMARY_SOFT_TRIGGER=0x06,
    OB_SYNC_MODE_SECONDARY_SOFT_TRIGGER=0x07,
    OB_SYNC_MODE_IR_IMU_SYNC=0x08, OB_SYNC_MODE_UNKNOWN=0xff
}

```

Sync mode. [More...](#)

```

enum OBDepthWorkModeTag { OB_DEVICE_DEPTH_WORK_MODE=0,
OB_CUSTOM_DEPTH_WORK_MODE=1 }

```

Preset tag. [More...](#)

enum **OBHoleFillingMode** { **OB\_HOLE\_FILL\_TOP** = 0 , **OB\_HOLE\_FILL\_NEAREST** = 1 ,  
**OB\_HOLE\_FILL\_FAREST** = 2 }  
Hole fillig mode. [More...](#)

enum **OB\_EDGE\_NOISE\_REMOVAL\_TYPE** { **OB\_MG\_FILTER** = 0 , **OB\_MGH\_FILTER** = 1 ,  
**OB\_MGA\_FILTER** = 2 , **OB\_MGC\_FILTER** = 3 }  
enum **OB\_DDO\_NOISE\_REMOVAL\_TYPE** { **OB\_NR\_LUT** = 0 , **OB\_NR\_OVERALL** = 1 }  
Denoising method. [More...](#)

enum **OB\_CMD\_VERSION** {  
    **OB\_CMD\_VERSION\_V0** = (uint16\_t)0 , **OB\_CMD\_VERSION\_V1** = (uint16\_t)1 ,  
    **OB\_CMD\_VERSION\_V2** = (uint16\_t)2 , **OB\_CMD\_VERSION\_V3** = (uint16\_t)3 ,  
    **OB\_CMD\_VERSION\_NOVERSION** = (uint16\_t)0xffffe , **OB\_CMD\_VERSION\_INVALID** =  
    (uint16\_t)0xffff  
}  
Command version associated with property id. [More...](#)

enum **OBCommunicationType** { **OB\_COMM\_USB** = 0x00 , **OB\_COMM\_NET** = 0x01 }  
Device communication mode. [More...](#)

enum **OBUSBPowerState** { **OB\_USB\_POWER\_NO\_PLUGIN** = 0 , **OB\_USB\_POWER\_5V\_0A9** = 1 ,  
**OB\_USB\_POWER\_5V\_1A5** = 2 , **OB\_USB\_POWER\_5V\_3A0** = 3 }  
USB power status. [More...](#)

enum **OBDCPowerState** { **OB\_DC\_POWER\_NO\_PLUGIN** = 0 , **OB\_DC\_POWER\_PLUGIN** = 1 }  
DC power status. [More...](#)

enum **ob\_rotate\_degree\_type** { **OB\_ROTATE\_DEGREE\_0** = 0 , **OB\_ROTATE\_DEGREE\_90** = 90 ,  
**OB\_ROTATE\_DEGREE\_180** = 180 , **OB\_ROTATE\_DEGREE\_270** = 270 }  
Rotate degree. [More...](#)

enum **ob\_power\_line\_freq\_mode** { **OB\_POWER\_LINE\_FREQ\_MODE\_CLOSE** = 0 ,  
**OB\_POWER\_LINE\_FREQ\_MODE\_50HZ** = 1 , **OB\_POWER\_LINE\_FREQ\_MODE\_60HZ** = 2  
}  
Power line frequency mode, for color camera anti-flicker configuration. [More...](#)

enum **OB\_FRAME\_AGGREGATE\_OUTPUT\_MODE** {  
    **OB\_FRAME\_AGGREGATE\_OUTPUT\_ALL\_TYPE\_FRAME\_REQUIRE** = 0 ,  
    **OB\_FRAME\_AGGREGATE\_OUTPUT\_COLOR\_FRAME\_REQUIRE** ,  
    **OB\_FRAME\_AGGREGATE\_OUTPUT\_ANY\_SITUATION** ,  
    **OB\_FRAME\_AGGREGATE\_OUTPUT\_DISABLE** }  
Frame aggregate output mode. [More...](#)

enum **OB\_COORDINATE\_SYSTEM\_TYPE** { **OB\_LEFT\_HAND\_COORDINATE\_SYSTEM** = 0 ,  
**OB\_RIGHT\_HAND\_COORDINATE\_SYSTEM** = 1 }  
Enumeration of point cloud coordinate system types. [More...](#)

enum **OB\_DEVICE DEVELOPMENT MODE** { **OB\_USER\_MODE** = 0 , **OB\_DEVELOPER\_MODE** =  
1 }  
Enumeration of device development modes. [More...](#)

```
enum ob_multi_device_sync_mode {
    OB_MULTI_DEVICE_SYNC_MODE_FREE_RUN= 1 << 0 ,
    OB_MULTI_DEVICE_SYNC_MODE_STANDALONE= 1 << 1 ,
    OB_MULTI_DEVICE_SYNC_MODE_PRIMARY = 1 << 2 ,
    OB_MULTI_DEVICE_SYNC_MODE_SECONDARY = 1 << 3 ,
    OB_MULTI_DEVICE_SYNC_MODE_SECONDARY_SYNCED = 1 << 4 ,
    OB_MULTI_DEVICE_SYNC_MODE_SOFTWARE_TRIGGERING= 1 << 5 ,
    OB_MULTI_DEVICE_SYNC_MODE_HARDWARE_TRIGGERING= 1 << 6 ,
    OB_MULTI_DEVICE_SYNC_MODE_IR_IMU_SYNC = 1 << 7
}
```

The synchronization mode of the device. [More...](#)

```
enum OBFilterConfigValueType { OB_FILTER_CONFIG_VALUE_TYPE_INVALID = -1 ,
    OB_FILTER_CONFIG_VALUE_TYPE_INT = 0 ,
    OB_FILTER_CONFIG_VALUE_TYPE_FLOAT = 1 ,
    OB_FILTER_CONFIG_VALUE_TYPE_BOOLEAN= 2 }
```

```

enum ob_frame_metadata_type {
    OB_FRAME_METADATA_TYPE_TIMESTAMP = 0 ,
    OB_FRAME_METADATA_TYPE_SENSOR_TIMESTAMP = 1 ,
    OB_FRAME_METADATA_TYPE_FRAME_NUMBER = 2 ,
    OB_FRAME_METADATA_TYPE_AUTO_EXPOSURE = 3 ,
    OB_FRAME_METADATA_TYPE_EXPOSURE = 4 , OB_FRAME_METADATA_TYPE_GAIN
= 5 , OB_FRAME_METADATA_TYPE_AUTO_WHITE_BALANCE = 6 ,
    OB_FRAME_METADATA_TYPE_WHITE_BALANCE = 7 ,
    OB_FRAME_METADATA_TYPE_BRIGHTNESS = 8 ,
    OB_FRAME_METADATA_TYPE_CONTRAST = 9 ,
    OB_FRAME_METADATA_TYPE_SATURATION = 10 ,
    OB_FRAME_METADATA_TYPE_SHARPNESS = 11 ,
    OB_FRAME_METADATA_TYPE_BACKLIGHT_COMPENSATION = 12 ,
    OB_FRAME_METADATA_TYPE_HUE = 13 , OB_FRAME_METADATA_TYPE_GAMMA =
14 , OB_FRAME_METADATA_TYPE_POWER_LINE_FREQUENCY = 15 ,
    OB_FRAME_METADATA_TYPE_LOW_LIGHT_COMPENSATION = 16 ,
    OB_FRAME_METADATA_TYPE_MANUAL_WHITE_BALANCE = 17 ,
    OB_FRAME_METADATA_TYPE_ACTUAL_FRAME_RATE = 18 ,
    OB_FRAME_METADATA_TYPE_FRAME_RATE = 19 ,
    OB_FRAME_METADATA_TYPE_AE_ROI_LEFT = 20 ,
    OB_FRAME_METADATA_TYPE_AE_ROI_TOP = 21 ,
    OB_FRAME_METADATA_TYPE_AE_ROI_RIGHT = 22 ,
    OB_FRAME_METADATA_TYPE_AE_ROI_BOTTOM = 23 ,
    OB_FRAME_METADATA_TYPE_EXPOSURE_PRIORITY = 24 ,
    OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_NAME = 25 ,
    OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_SIZE = 26 ,
    OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_INDEX = 27 ,
    OB_FRAME_METADATA_TYPE_LASER_POWER = 28 ,
    OB_FRAME_METADATA_TYPE_LASER_POWER_LEVEL = 29 ,
    OB_FRAME_METADATA_TYPE_LASER_STATUS = 30 ,
    OB_FRAME_METADATA_TYPE_GPIO_INPUT_DATA = 31 ,
    OB_FRAME_METADATA_TYPE_DISPARITY_SEARCH_OFFSET = 32 ,
    OB_FRAME_METADATA_TYPE_DISPARITY_SEARCH_RANGE = 33 ,
    OB_FRAME_METADATA_TYPE_COUNT
}

```

Frame metadata types. [More...](#)

```

enum ob_uvc_backend_type { OB_UVC_BACKEND_TYPE_AUTO ,
OB_UVC_BACKEND_TYPE_LIBUVC , OB_UVC_BACKEND_TYPE_V4L2 ,
OB_UVC_BACKEND_TYPE_MSMF }

```

For Linux, there are two ways to access the UVC device, libuvc and v4l2. The backend type is used to select the backend to access the device. [More...](#)

```
enum ob_playback_status {
    OB_PLAYBACK_UNKNOWN, OB_PLAYBACK_PLAYING, OB_PLAYBACK_PAUSED,
    OB_PLAYBACK_STOPPED,
    OB_PLAYBACK_COUNT
}
```

The playback status of the media. More...

## Detailed Description

Provide structs commonly used in the SDK, enumerating constant definitions.

Definition in file **ObTypes.h**.

## Macro Definition Documentation

### ◆ OB\_WIDTH\_ANY

```
#define OB_WIDTH_ANY 0
```

Definition at line **44** of file **ObTypes.h**.

Referenced by **ob::Config::enableVideoStream()**, **ob::Config::enableVideoStream()**, and **ob::StreamProfileList::getVideoStreamProfile()**.

### ◆ OB\_HEIGHT\_ANY

```
#define OB_HEIGHT_ANY 0
```

Definition at line **45** of file **ObTypes.h**.

Referenced by **ob::Config::enableVideoStream()**, **ob::Config::enableVideoStream()**, and **ob::StreamProfileList::getVideoStreamProfile()**.

### ◆ OB\_FPS\_ANY

```
#define OB_FPS_ANY 0
```

Definition at line [46](#) of file **ObTypes.h**.

Referenced by **ob::Config::enableVideoStream()**, **ob::Config::enableVideoStream()**, and **ob::StreamProfileList::getVideoStreamProfile()**.

◆ **OB\_FORMAT\_ANY**

```
#define OB_FORMAT_ANY OB_FORMAT_UNKNOWN
```

Definition at line [47](#) of file **ObTypes.h**.

Referenced by **ob::Config::enableVideoStream()**, **ob::Config::enableVideoStream()**, and **ob::StreamProfileList::getVideoStreamProfile()**.

◆ **OB\_PROFILE\_DEFAULT**

```
#define OB_PROFILE_DEFAULT 0
```

Definition at line [48](#) of file **ObTypes.h**.

◆ **OB\_DEFAULT\_STRIDE\_BYTES**

```
#define OB_DEFAULT_STRIDE_BYTES 0
```

Definition at line [49](#) of file **ObTypes.h**.

◆ **OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY**

```
#define OB_ACCEL_FULL_SCALE_RANGE_ANY OB_ACCEL_FS_UNKNOWN
```

Definition at line [50](#) of file **ObTypes.h**.

Referenced by **ob::Config::enableAccelStream()**.

◆ **OB\_ACCEL\_SAMPLE\_RATE\_ANY**

```
#define OB_ACCEL_SAMPLE_RATE_ANY OB_SAMPLE_RATE_UNKNOWN
```

Definition at line [51](#) of file [ObTypes.h](#).

Referenced by [ob::Config::enableAccelStream\(\)](#).

◆ [OB\\_GYRO\\_FULL\\_SCALE\\_RANGE\\_ANY](#)

```
#define OB_GYRO_FULL_SCALE_RANGE_ANY OB_GYRO_FS_UNKNOWN
```

Definition at line [52](#) of file [ObTypes.h](#).

Referenced by [ob::Config::enableGyroStream\(\)](#).

◆ [OB\\_GYRO\\_SAMPLE\\_RATE\\_ANY](#)

```
#define OB_GYRO_SAMPLE_RATE_ANY OB_SAMPLE_RATE_UNKNOWN
```

Definition at line [53](#) of file [ObTypes.h](#).

Referenced by [ob::Config::enableGyroStream\(\)](#).

◆ [OB\\_PATH\\_MAX](#)

```
#define OB_PATH_MAX (1024)
```

maximum path length

Definition at line [58](#) of file [ObTypes.h](#).

Referenced by [ob\\_device\\_update\\_optional\\_depth\\_presets\(\)](#), and  
[ob::Device::updateOptionalDepthPresets\(\)](#).

◆ [OB\\_LOG\\_SEVERITY\\_NONE](#)

```
#define OB_LOG_SEVERITY_NONE OB_LOG_SEVERITY_OFF
```

Definition at line [93](#) of file [ObTypes.h](#).

◆ OB\_FORMAT\_RGB888

```
#define OB_FORMAT_RGB888 OB_FORMAT_RGB
```

Definition at line 238 of file **ObTypes.h**.

◆ OB\_FORMAT\_MJPEG

```
#define OB_FORMAT_MJPEG OB_FORMAT_MJPG
```

Definition at line 239 of file **ObTypes.h**.

◆ IS\_FIXED\_SIZE\_FORMAT

```
#define IS_FIXED_SIZE_FORMAT ( format )
```

**Value:**

```
(format != OB_FORMAT_MJPG && format != OB_FORMAT_H264 && format != OB_FORMAT_H265 &&
format != OB_FORMAT_HEVC && format != OB_FORMAT_RLE \
&& format != OB_FORMAT_RVL)
```

Definition at line 242 of file **ObTypes.h**.

◆ IS\_PACKED\_FORMAT

```
#define IS_PACKED_FORMAT ( format )
```

**Value:**

```
(format == OB_FORMAT_Y10 || format == OB_FORMAT_Y11 || format == OB_FORMAT_Y12 || format ==
OB_FORMAT_Y14 || format == OB_FORMAT_RLE)
```

Definition at line 247 of file **ObTypes.h**.

◆ FORMAT\_MJPEG\_TO\_I420

```
#define FORMAT_MJPEG_TO_I420 FORMAT_MJPG_TO_I420
```

Definition at line 562 of file **ObTypes.h**.

◆ FORMAT\_MJPEG\_TO\_NV21

```
#define FORMAT_MJPEG_TO_NV21 FORMAT_MJPG_TO_NV21
```

Definition at line **563** of file **ObTypes.h**.

◆ FORMAT\_MJPEG\_TO\_BGRA

```
#define FORMAT_MJPEG_TO_BGRA FORMAT_MJPG_TO_BGRA
```

Definition at line **564** of file **ObTypes.h**.

◆ FORMAT\_YUYV\_TO\_RGB888

```
#define FORMAT_YUYV_TO_RGB888 FORMAT_YUV_TO_RGB
```

Definition at line **565** of file **ObTypes.h**.

◆ FORMAT\_I420\_TO\_RGB888

```
#define FORMAT_I420_TO_RGB888 FORMAT_I420_TO_RGB
```

Definition at line **566** of file **ObTypes.h**.

◆ FORMAT\_NV21\_TO\_RGB888

```
#define FORMAT_NV21_TO_RGB888 FORMAT_NV21_TO_RGB
```

Definition at line **567** of file **ObTypes.h**.

◆ FORMAT\_NV12\_TO\_RGB888

```
#define FORMAT_NV12_TO_RGB888 FORMAT_NV12_TO_RGB
```

Definition at line **568** of file **ObTypes.h**.

◆ FORMAT\_UYVY\_TO\_RGB888

```
#define FORMAT_UYVY_TO_RGB888 FORMAT_UYVY_TO_RGB
```

Definition at line **569** of file **ObTypes.h**.

◆ FORMAT\_MJPG\_TO\_RGB888

```
#define FORMAT_MJPG_TO_RGB888 FORMAT_MJPG_TO_RGB
```

Definition at line **570** of file **ObTypes.h**.

◆ FORMAT\_MJPG\_TO\_BGR888

```
#define FORMAT_MJPG_TO_BGR888 FORMAT_MJPG_TO_BGR
```

Definition at line **571** of file **ObTypes.h**.

◆ FORMAT\_MJPEG\_TO\_RGB888

```
#define FORMAT_MJPEG_TO_RGB888 FORMAT_MJPEG_TO_RGB
```

Definition at line **572** of file **ObTypes.h**.

◆ FORMAT\_MJPEG\_TO\_BGR888

```
#define FORMAT_MJPEG_TO_BGR888 FORMAT_MJPEG_TO_BGR
```

Definition at line **573** of file **ObTypes.h**.

◆ FORMAT\_RGB888\_TO\_BGR

```
#define FORMAT_RGB888_TO_BGR FORMAT_RGB_TO_BGR
```

Definition at line **574** of file **ObTypes.h**.

◆ OBDeviceIpAddrConfig

```
#define OBDeviceIpAddrConfig OBNetIpConfig
```

Definition at line 1112 of file **ObTypes.h**.

◆ ob\_device\_ip\_addr\_config

```
#define ob_device_ip_addr_config OBNetIpConfig
```

Definition at line 1113 of file **ObTypes.h**.

◆ OB\_FRAME\_AGGREGATE\_OUTPUT\_FULL\_FRAME\_REQUIRE

```
#define  
OB_FRAME_AGGREGATE_OUTPUT_FULL_FRAME_REQUIRE OB_FRAME_AGGREGATE_OUTPUT_ALL_TY
```

Definition at line 1196 of file **ObTypes.h**.

◆ OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER\_MODE

```
#define  
OB_FRAME_METADATA_TYPE_LASER_POWER_MODE OB_FRAME_METADATA_TYPE_LASER_POWER_
```

Definition at line 1730 of file **ObTypes.h**.

◆ OB\_FRAME\_METADATA\_TYPE\_EMITTER\_MODE

```
#define  
OB_FRAME_METADATA_TYPE_EMITTER_MODE OB_FRAME_METADATA_TYPE_LASER_STATUS
```

Definition at line 1731 of file **ObTypes.h**.

◆ ob\_filter\_callback

```
#define ob_filter_callback ob_frame_callback
```

Definition at line **1807** of file **ObTypes.h**.

Referenced by **ob\_filter\_set\_callback()**.

◆ **ob\_playback\_callback**

```
#define ob_playback_callback ob_frame_callback
```

Definition at line **1808** of file **ObTypes.h**.

◆ **ob\_is\_video\_sensor\_type**

```
#define ob_is_video_sensor_type ( sensor_type )
```

**Value:**

```
(sensor_type == OB_SENSOR_COLOR || sensor_type == OB_SENSOR_DEPTH || sensor_type == OB_SENSOR_IR || sensor_type == OB_SENSOR_IR_LEFT \|| sensor_type == OB_SENSOR_IR_RIGHT || sensor_type == OB_SENSOR_CONFIDENCE)
```

Check if the sensor\_type is a video sensor.

**Parameters**

**sensor\_type** Sensor type to check

**Returns**

True if sensor\_type is a video sensor, false otherwise

Definition at line **1842** of file **ObTypes.h**.

Referenced by **ob::TypeHelper::isVideoSensorType()**.

◆ **ob\_is\_video\_stream\_type**

```
#define ob_is_video_stream_type ( stream_type )
```

**Value:**

```
(stream_type == OB_STREAM_COLOR || stream_type == OB_STREAM_DEPTH || stream_type == OB_ST  
REAM_IR || stream_type == OB_STREAM_IR_LEFT \  
|| stream_type == OB_STREAM_IR_RIGHT || stream_type == OB_STREAM_VIDEO || stream_type == OB_  
STREAM_CONFIDENCE)
```

check if the stream\_type is a video stream

**Parameters**

**stream\_type** Stream type to check

**Returns**

True if stream\_type is a video stream, false otherwise

Definition at line [1852](#) of file **ObTypes.h**.

Referenced by **ob::TypeHelper::isVideoStreamType()**.

◆ **is\_ir\_sensor**

```
#define is_ir_sensor ( sensor_type )
```

**Value:**

```
(sensor_type == OB_SENSOR_IR || sensor_type == OB_SENSOR_IR_LEFT || sensor_type == OB_SENSOR_I  
R_RIGHT)
```

Check if sensor\_type is an IR sensor.

**Parameters**

**sensor\_type** Sensor type to check

**Returns**

True if sensor\_type is an IR sensor, false otherwise

Definition at line [1862](#) of file **ObTypes.h**.

◆ **isIRSensor**

```
#define isIRSensor is_ir_sensor
```

Definition at line [1863](#) of file **ObTypes.h**.

◆ **is\_ir\_stream**

```
#define is_ir_stream( stream_type )
```

**Value:**

```
(stream_type == OB_STREAM_IR || stream_type == OB_STREAM_IR_LEFT || stream_type == OB_STREAM_IR_RIGHT)
```

Check if stream\_type is an IR stream.

**Parameters**

**stream\_type** Stream type to check

**Returns**

True if stream\_type is an IR stream, false otherwise

Definition at line **1871** of file **ObTypes.h**.

◆ **isIRStream**

```
#define isIRStream is_ir_stream
```

Definition at line **1872** of file **ObTypes.h**.

◆ **is\_ir\_frame**

```
#define is_ir_frame( frame_type )
```

**Value:**

```
(frame_type == OB_FRAME_IR || frame_type == OB_FRAME_IR_LEFT || frame_type == OB_FRAME_IR_RIGHT)
```

Check if frame\_type is an IR frame.

**Parameters**

**frame\_type** Frame type to check

**Returns**

True if frame\_type is an IR frame, false otherwise

Definition at line **1880** of file **ObTypes.h**.

◆ **isIRFrame**

```
#define isIRFrame is_ir_frame
```

Definition at line **1881** of file **ObTypes.h**.

◆ **OB\_DEFAULT\_DECRYPT\_KEY**

```
#define OB_DEFAULT_DECRYPT_KEY (nullptr)
```

The default Decrypt Key.

Definition at line **1886** of file **ObTypes.h**.

## Typedef Documentation

◆ **ob\_context**

```
typedef struct ob_context_t ob_context
```

Definition at line **22** of file **ObTypes.h**.

◆ **ob\_device**

```
typedef struct ob_device_t ob_device
```

Definition at line **23** of file **ObTypes.h**.

◆ **ob\_device\_info**

```
typedef struct ob_device_info_t ob_device_info
```

Definition at line **24** of file **ObTypes.h**.

◆ **ob\_device\_list**

```
typedef struct ob_device_list_t ob_device_list
```

Definition at line **25** of file **ObTypes.h**.

- ◆ **ob\_record\_device**

```
typedef struct ob_record_device_t ob_record_device
```

Definition at line **26** of file **ObTypes.h**.

- ◆ **ob\_playback\_device**

```
typedef struct ob_playback_device_t ob_playback_device
```

Definition at line **27** of file **ObTypes.h**.

- ◆ **ob\_camera\_param\_list**

```
typedef struct ob_camera_param_list_t ob_camera_param_list
```

Definition at line **28** of file **ObTypes.h**.

- ◆ **ob\_sensor**

```
typedef struct ob_sensor_t ob_sensor
```

Definition at line **29** of file **ObTypes.h**.

- ◆ **ob\_sensor\_list**

```
typedef struct ob_sensor_list_t ob_sensor_list
```

Definition at line **30** of file **ObTypes.h**.

- ◆ **ob\_stream\_profile**

```
typedef struct ob_stream_profile_t ob_stream_profile
```

Definition at line [31](#) of file **ObTypes.h**.

- ◆ **ob\_stream\_profile\_list**

```
typedef struct ob_stream_profile_list_t ob_stream_profile_list
```

Definition at line [32](#) of file **ObTypes.h**.

- ◆ **ob\_frame**

```
typedef struct ob_frame_t ob_frame
```

Definition at line [33](#) of file **ObTypes.h**.

- ◆ **ob\_filter**

```
typedef struct ob_filter_t ob_filter
```

Definition at line [34](#) of file **ObTypes.h**.

- ◆ **ob\_filter\_list**

```
typedef struct ob_filter_list_t ob_filter_list
```

Definition at line [35](#) of file **ObTypes.h**.

- ◆ **ob\_pipeline**

```
typedef struct ob_pipeline_t ob_pipeline
```

Definition at line [36](#) of file **ObTypes.h**.

- ◆ **ob\_config**

```
typedef struct ob_config_t ob_config
```

Definition at line [37](#) of file **ObTypes.h**.

- ◆ [\*\*ob\\_depth\\_work\\_mode\\_list\*\*](#)

```
typedef struct ob_depth_work_mode_list_t ob_depth_work_mode_list
```

Definition at line [38](#) of file **ObTypes.h**.

- ◆ [\*\*ob\\_device\\_preset\\_list\*\*](#)

```
typedef struct ob_device_preset_list_t ob_device_preset_list
```

Definition at line [39](#) of file **ObTypes.h**.

- ◆ [\*\*ob\\_filter\\_config\\_schema\\_list\*\*](#)

```
typedef struct ob_filter_config_schema_list_t ob_filter_config_schema_list
```

Definition at line [40](#) of file **ObTypes.h**.

- ◆ [\*\*ob\\_device\\_frame\\_interleave\\_list\*\*](#)

```
typedef struct ob_device_frame_interleave_list_t ob_device_frame_interleave_list
```

Definition at line [41](#) of file **ObTypes.h**.

- ◆ [\*\*ob\\_preset\\_resolution\\_config\\_list\*\*](#)

```
typedef struct ob_preset_resolution_config_list_t ob_preset_resolution_config_list
```

Definition at line [42](#) of file **ObTypes.h**.

- ◆ [\*\*ob\\_permission\\_type\*\*](#)

typedef enum **OBPermissionType** **ob\_permission\_type**

◆ **ob\_status**

typedef enum **OBStatus** **ob\_status**

◆ **ob\_log\_severity**

typedef enum **OBLogSeverity** **ob\_log\_severity**

◆ **DEVICE\_LOG\_SEVERITY\_LEVEL**

typedef enum **OBLogSeverity** **DEVICE\_LOG\_SEVERITY\_LEVEL**

◆ **OBDeviceLogSeverityLevel**

typedef enum **OBLogSeverity** **OBDeviceLogSeverityLevel**

◆ **ob\_device\_log\_severity\_level**

typedef enum **OBLogSeverity** **ob\_device\_log\_severity\_level**

◆ **ob\_exception\_type**

typedef enum **OBExceptionType** **ob\_exception\_type**

◆ **ob\_error**

typedef struct ob\_error **ob\_error**

The error class exposed by the SDK, users can get detailed error information according to the error.

◆ **ob\_sensor\_type**

typedef enum **OBSensorType** **ob\_sensor\_type**

◆ ob\_stream\_type

typedef enum **OBStreamType** ob\_stream\_type

◆ ob\_frame\_type

typedef enum **OBFrameType** ob\_frame\_type

◆ ob\_pixel\_type

typedef enum **OBPixelType** ob\_pixel\_type

◆ ob\_format

typedef enum **OBFormat** ob\_format

◆ OBFwUpdateState

typedef enum **OBUpgradeState** OBFwUpdateState

◆ ob\_upgrade\_state

typedef enum **OBUpgradeState** ob\_upgrade\_state

◆ ob\_fw\_update\_state

typedef enum **OBUpgradeState** ob\_fw\_update\_state

◆ ob\_file\_tran\_state

typedef enum **OBFileTranState** ob\_file\_tran\_state

◆ ob\_data\_tran\_state

typedef enum **OBDataTranState** ob\_data\_tran\_state

◆ `ob_data_chunk`

`typedef struct OBDataChunk ob_data_chunk`

◆ `ob_int_property_range`

`typedef struct OBIntPropertyRange ob_int_property_range`

◆ `ob_float_property_range`

`typedef struct OBFloatPropertyRange ob_float_property_range`

◆ `ob_uint16_property_range`

`typedef struct OBUInt16PropertyRange ob_uint16_property_range`

◆ `ob_uint8_property_range`

`typedef struct OBUInt8PropertyRange ob_uint8_property_range`

◆ `ob_bool_property_range`

`typedef struct OBBBoolPropertyRange ob_bool_property_range`

◆ `ob_camera_distortion_model`

`typedef enum OBCameraDistortionModel ob_camera_distortion_model`

◆ `ob_camera_intrinsic`

`typedef struct OBCameraIntrinsic ob_camera_intrinsic`

◆ `ob_accel_intrinsic`

`typedef struct OBAccelIntrinsic ob_accel_intrinsic`

- ◆ ob\_gyro\_intrinsic

```
typedef struct OBGyroIntrinsic ob_gyro_intrinsic
```

- ◆ ob\_camera\_distortion

```
typedef struct OBCameraDistortion ob_camera_distortion
```

- ◆ ob\_d2c\_transform

```
typedef struct OBD2CTransform ob_d2c_transform
```

- ◆ OBTransform

```
typedef struct OBD2CTransform OBTransform
```

- ◆ ob\_transform

```
typedef struct OBD2CTransform ob_transform
```

- ◆ OBExtrinsic

```
typedef struct OBD2CTransform OBExtrinsic
```

- ◆ ob\_extrinsic

```
typedef struct OBD2CTransform ob_extrinsic
```

- ◆ ob\_camera\_param

```
typedef struct OBCameraParam ob_camera_param
```

- ◆ ob\_preset\_resolution\_ratio\_config

```
typedef struct OBPresetResolutionConfig ob_preset_resolution_ratio_config
```

◆ ob\_calibration\_param

typedef struct **OBCalibrationParam** ob\_calibration\_param

◆ OBMarginFilterConfig

typedef struct **ob\_margin\_filter\_config** OBMarginFilterConfig

◆ ob\_mgc\_filter\_config

typedef struct **OBMGCFConfig** ob\_mgc\_filter\_config

◆ ob\_align\_mode

typedef enum **OBAAlignMode** ob\_align\_mode

◆ ob\_camera\_performance\_mode

typedef enum **OBCameraPerformanceMode** ob\_camera\_performance\_mode

◆ ob\_rect

typedef struct **OBRect** ob\_rect

◆ ob\_convert\_format

typedef enum **OBConvertFormat** ob\_convert\_format

◆ OBGyroSampleRate

typedef enum **OBIMUSampleRate** OBGyroSampleRate

◆ ob\_gyro\_sample\_rate

typedef enum **OBIMUSampleRate** ob\_gyro\_sample\_rate

◆ OBAccelSampleRate

typedef enum **OBIMUSampleRate** OBAccelSampleRate

◆ ob\_accel\_sample\_rate

typedef enum **OBIMUSampleRate** ob\_accel\_sample\_rate

◆ OB\_SAMPLE\_RATE

typedef enum **OBIMUSampleRate** OB\_SAMPLE\_RATE

◆ ob\_gyro\_full\_scale\_range

typedef enum **OBGyroFullScaleRange** ob\_gyro\_full\_scale\_range

◆ OB\_GYRO\_FULL\_SCALE\_RANGE

typedef enum **OBGyroFullScaleRange** OB\_GYRO\_FULL\_SCALE\_RANGE

◆ ob\_accel\_full\_scale\_range

typedef enum **OBAccelFullScaleRange** ob\_accel\_full\_scale\_range

◆ OB\_ACCEL\_FULL\_SCALE\_RANGE

typedef enum **OBAccelFullScaleRange** OB\_ACCEL\_FULL\_SCALE\_RANGE

◆ OBGyroValue

typedef struct **OBAccelValue** OBGyroValue

◆ OBFloat3D

typedef struct **OBAccelValue** OBFloat3D

◆ **ob\_accel\_value**

typedef struct **OBAccelValue** **ob\_accel\_value**

◆ **ob\_gyro\_value**

typedef struct **OBAccelValue** **ob\_gyro\_value**

◆ **ob\_float\_3d**

typedef struct **OBAccelValue** **ob\_float\_3d**

◆ **OBDeviceState**

typedef uint64\_t **OBDeviceState**

Device state.

Definition at line **647** of file **ObTypes.h**.

◆ **ob\_device\_state**

typedef uint64\_t **ob\_device\_state**

Definition at line **647** of file **ObTypes.h**.

◆ **ob\_device\_temperature**

typedef struct **OBDeviceTemperature** **ob\_device\_temperature**

◆ **DEVICE\_TEMPERATURE**

typedef struct **OBDeviceTemperature** **DEVICE\_TEMPERATURE**

◆ **ob\_depth\_cropping\_mode**

typedef enum **OBDepthCroppingMode** **ob\_depth\_cropping\_mode**

◆ OB\_DEPTH\_CROPPING\_MODE

typedef enum **OBDepthCroppingMode** **OB\_DEPTH\_CROPPING\_MODE**

◆ ob\_device\_type

typedef enum **OBDeviceType** **ob\_device\_type**

◆ OB\_DEVICE\_TYPE

typedef enum **OBDeviceType** **OB\_DEVICE\_TYPE**

◆ ob\_media\_type

typedef enum **OBMediaType** **ob\_media\_type**

◆ OB\_MEDIA\_TYPE

typedef enum **OBMediaType** **OB\_MEDIA\_TYPE**

◆ ob\_media\_state

typedef enum **OBMediaState** **ob\_media\_state**

◆ OB\_MEDIA\_STATE\_EM

typedef enum **OBMediaState** **OB\_MEDIA\_STATE\_EM**

◆ ob\_depth\_precision\_level

typedef enum **OBDepthPrecisionLevel** **ob\_depth\_precision\_level**

◆ OB\_DEPTH\_PRECISION\_LEVEL

typedef enum **OBDepthPrecisionLevel** OB\_DEPTH\_PRECISION\_LEVEL

◆ OBDepthUnit

typedef enum **OBDepthPrecisionLevel** OBDepthUnit

◆ ob\_depth\_unit

typedef enum **OBDepthPrecisionLevel** ob\_depth\_unit

◆ ob\_disparity\_param

typedef struct **OBDisparityParam** ob\_disparity\_param

◆ ob\_tof\_filter\_range

typedef enum **OBToFilterRange** ob\_tof\_filter\_range

◆ TOF\_FILTER\_RANGE

typedef enum **OBToFilterRange** TOF\_FILTER\_RANGE

◆ ob\_point

typedef struct **OBPoint** ob\_point

◆ OBPoint3f

typedef struct **OBPoint** OBPoint3f

◆ ob\_point3f

typedef struct **OBPoint** ob\_point3f

- ◆ **ob\_point2f**

typedef struct **OBPoint2f** **ob\_point2f**

- ◆ **ob\_xy\_tables**

typedef struct **OBXYTables** **ob\_xy\_tables**

- ◆ **ob\_color\_point**

typedef struct **OBCColorPoint** **ob\_color\_point**

- ◆ **ob\_compression\_mode**

typedef enum **OBCompressionMode** **ob\_compression\_mode**

- ◆ **OB\_COMPRESSION\_MODE**

typedef enum **OBCompressionMode** **OB\_COMPRESSION\_MODE**

- ◆ **ob\_compression\_params**

typedef struct **OBCompressionParams** **ob\_compression\_params**

- ◆ **OB\_COMPRESSION\_PARAMS**

typedef struct **OBCompressionParams** **OB\_COMPRESSION\_PARAMS**

- ◆ **ob\_tof\_exposure\_threshold\_control**

typedef struct **OBTofExposureThresholdControl** **ob\_tof\_exposure\_threshold\_control**

- ◆ **TOF\_EXPOSURE\_THRESHOLD\_CONTROL**

typedef struct **OBTofExposureThresholdControl** **TOF\_EXPOSURE\_THRESHOLD\_CONTROL**

- ◆ ob\_sync\_mode

```
typedef enum OBSyncMode ob_sync_mode
```

- ◆ OB\_SYNC\_MODE

```
typedef enum OBSyncMode OB_SYNC_MODE
```

- ◆ ob\_device\_sync\_config

```
typedef struct OBDeviceSyncConfig ob_device_sync_config
```

- ◆ OB\_DEVICE\_SYNC\_CONFIG

```
typedef struct OBDeviceSyncConfig OB_DEVICE_SYNC_CONFIG
```

- ◆ ob\_depth\_work\_mode\_tag

```
typedef enum OBDepthWorkModeTag ob_depth_work_mode_tag
```

- ◆ ob\_depth\_work\_mode

```
typedef struct OBDepthWorkMode ob_depth_work_mode
```

- ◆ ob\_sequence\_id\_item

```
typedef struct OBSequenceIdItem ob_sequence_id_item
```

- ◆ ob\_hole\_filling\_mode

```
typedef enum OBHoleFillingMode ob_hole_filling_mode
```

- ◆ ob\_spatial\_advanced\_filter\_params

```
typedef struct OBSpatialAdvancedFilterParams ob_spatial_advanced_filter_params
```

- ◆ OBEdgeNoiseRemovalType

```
typedef enum OB_EDGE_NOISE_REMOVAL_TYPE OBEdgeNoiseRemovalType
```

- ◆ ob\_edge\_noise\_removal\_type

```
typedef enum OB_EDGE_NOISE_REMOVAL_TYPE ob_edge_noise_removal_type
```

- ◆ ob\_edge\_noise\_removal\_filter\_params

```
typedef struct OBEdgeNoiseRemovalFilterParams ob_edge_noise_removal_filter_params
```

- ◆ OBDDONoiseRemovalType

```
typedef enum OB_DDO_NOISE_REMOVAL_TYPE OBDDONoiseRemovalType
```

Denoising method.

- ◆ ob\_ddo\_noise\_removal\_type

```
typedef enum OB_DDO_NOISE_REMOVAL_TYPE ob_ddo_noise_removal_type
```

- ◆ ob\_noise\_removal\_filter\_params

```
typedef struct OBNoiseRemovalFilterParams ob_noise_removal_filter_params
```

- ◆ ob\_protocol\_version

```
typedef struct OBProtocolVersion ob_protocol_version
```

- ◆ OBCmdVersion

```
typedef enum OB_CMD_VERSION OBCmdVersion
```

- ◆ ob\_cmd\_version

`typedef enum OB_CMD_VERSION ob_cmd_version`

◆ `ob_net_ip_config`

`typedef struct OBNetIpConfig ob_net_ip_config`

◆ `DEVICE_IP_ADDR_CONFIG`

`typedef struct OBNetIpConfig DEVICE_IP_ADDR_CONFIG`

◆ `ob_communication_type`

`typedef enum OBCommunicationType ob_communication_type`

◆ `OB_COMMUNICATION_TYPE`

`typedef enum OBCommunicationType OB_COMMUNICATION_TYPE`

◆ `ob_usb_power_state`

`typedef enum OBUSBPowerState ob_usb_power_state`

◆ `ob_dc_power_state`

`typedef enum OBDCPowerState ob_dc_power_state`

◆ `OBRotateDegreeType`

`typedef enum ob_rotate_degree_type OBRotateDegreeType`

◆ `OBPowerLineFreqMode`

`typedef enum ob_power_line_freq_mode OBPowerLineFreqMode`

- ◆ OBFrameAggregateOutputMode

```
typedef enum OB_FRAME_AGGREGATE_OUTPUT_MODE OBFrameAggregateOutputMode
```

- ◆ ob\_frame\_aggregate\_output\_mode

```
typedef enum OB_FRAME_AGGREGATE_OUTPUT_MODE ob_frame_aggregate_output_mode
```

- ◆ OBCoordinateSystemType

```
typedef enum OB_COORDINATE_SYSTEM_TYPE OBCoordinateSystemType
```

- ◆ ob\_coordinate\_system\_type

```
typedef enum OB_COORDINATE_SYSTEM_TYPE ob_coordinate_system_type
```

- ◆ OBDeviceDevelopmentMode

```
typedef enum OB_DEVICE DEVELOPMENT MODE OBDeviceDevelopmentMode
```

- ◆ ob\_device\_development\_mode

```
typedef enum OB_DEVICE DEVELOPMENT MODE ob_device_development_mode
```

- ◆ OBMultiDeviceSyncMode

```
typedef enum ob_multi_device_sync_mode OBMultiDeviceSyncMode
```

- ◆ OBMultiDeviceSyncConfig

```
typedef struct ob_multi_device_sync_config OBMultiDeviceSyncConfig
```

- ◆ OBDeviceTimestampResetConfig

```
typedef struct ob_device_timestamp_reset_config OBDeviceTimestampResetConfig
```

- ◆ ob\_baseline\_calibration\_param

```
typedef struct BASELINE_CALIBRATION_PARAM ob_baseline_calibration_param
```

- ◆ OBBaselineCalibrationParam

```
typedef struct BASELINE_CALIBRATION_PARAM OBBaselineCalibrationParam
```

- ◆ ob\_hdr\_config

```
typedef struct HDR_CONFIG ob_hdr_config
```

- ◆ OBHdrConfig

```
typedef struct HDR_CONFIG OBHdrConfig
```

- ◆ ob\_region\_of\_interest

```
typedef struct AE_ROI ob_region_of_interest
```

- ◆ OBRegionOfInterest

```
typedef struct AE_ROI OBRegionOfInterest
```

- ◆ ob\_filter\_config\_value\_type

```
typedef enum OBFilterConfigValueType ob_filter_config_value_type
```

- ◆ ob\_filter\_config\_schema\_item

```
typedef struct OBFilterConfigSchemaItem ob_filter_config_schema_item
```

- ◆ ob\_device\_serial\_number

```
typedef struct OBDeviceSerialNumber ob_device_serial_number
```

- ◆ OBSerialNumber

```
typedef struct OBDeviceSerialNumber OBSerialNumber
```

- ◆ ob\_serial\_number

```
typedef struct OBDeviceSerialNumber ob_serial_number
```

- ◆ ob\_disp\_offset\_config

```
typedef struct OBDispOffsetConfig ob_disp_offset_config
```

- ◆ OBFrameMetadataType

```
typedef enum ob_frame_metadata_type OBFrameMetadataType
```

- ◆ OBUvcBackendType

```
typedef enum ob_uvc_backend_type OBUvcBackendType
```

- ◆ OBPlaybackStatus

```
typedef enum ob_playback_status OBPlaybackStatus
```

- ◆ ob\_file\_send\_callback

```
typedef void(* ob_file_send_callback)(ob_file_tran_state state, const char *message, uint8_t percent, void *user_data)
```

Callback for file transfer.

#### Parameters

- state** Transmission status
- message** Transfer status information
- percent** Transfer progress percentage
- user\_data** User-defined data

Definition at line [1741](#) of file [ObTypes.h](#).

### ◆ ob\_device\_fw\_update\_callback

```
typedef void(* ob_device_fw_update_callback)(ob_fw_update_state state, const char *message, uint8_t percent, void *user_data)
```

Callback for firmware upgrade.

#### Parameters

- state** Upgrade status
- message** Upgrade status information
- percent** Upgrade progress percentage
- user\_data** User-defined data

Definition at line [1751](#) of file [ObTypes.h](#).

### ◆ ob\_device\_state\_callback

```
typedef void(* ob_device_state_callback)(ob_device_state state, const char *message, void *user_data)
```

Callback for device status.

#### Parameters

- state** Device status
- message** Device status information
- user\_data** User-defined data

Definition at line [1760](#) of file [ObTypes.h](#).

## ◆ ob\_set\_data\_callback

```
typedef void(* ob_set_data_callback) (ob_data_tran_state state, uint8_t percent, void *user_data)
```

Callback for writing data.

### Parameters

**state** Write data status

**percent** Write data percentage

**user\_data** User-defined data

Definition at line [1769](#) of file **ObTypes.h**.

## ◆ ob\_get\_data\_callback

```
typedef void(* ob_get_data_callback) (ob_data_tran_state state, ob_data_chunk *dataChunk, void *user_data)
```

Callback for reading data.

### Parameters

**state** Read data status

**dataChunk** Read the returned data block

**user\_data** User-defined data

Definition at line [1778](#) of file **ObTypes.h**.

## ◆ ob\_media\_state\_callback

```
typedef void(* ob_media_state_callback) (ob_media_state state, void *user_data)
```

Callback for media status (recording and playback)

### Parameters

**state** Condition

**user\_data** User-defined data

Definition at line [1786](#) of file **ObTypes.h**.

## ◆ ob\_device\_changed\_callback

```
typedef void(* ob_device_changed_callback)(ob_device_list *removed, ob_device_list *added, void *user_data)
```

Callback for device change.

#### Parameters

**removed** List of deleted (dropped) devices

**added** List of added (online) devices

**user\_data** User-defined data

Definition at line [1795](#) of file [ObTypes.h](#).

### ◆ [ob\\_frame\\_callback](#)

```
typedef void(* ob_frame_callback)(ob_frame *frame, void *user_data)
```

Callback for frame.

#### Parameters

**frame** Frame object

**user\_data** User-defined data

Definition at line [1806](#) of file [ObTypes.h](#).

### ◆ [ob\\_frameset\\_callback](#)

```
typedef void(* ob_frameset_callback)(ob_frame *frameset, void *user_data)
```

Callback for frameset.

#### Parameters

**frameset** Frameset object

**user\_data** User-defined data

Definition at line [1816](#) of file [ObTypes.h](#).

### ◆ [ob\\_frame\\_destroy\\_callback](#)

```
typedef void ob_frame_destroy_callback(uint8_t *buffer, void *user_data)
```

Customize the delete callback.

#### Parameters

- buffer** Data that needs to be deleted
- user\_data** User-defined data

Definition at line **1824** of file **ObTypes.h**.

#### ◆ **ob\_log\_callback**

```
typedef void ob_log_callback(ob_log_severity severity, const char *message, void *user_data)
```

Callback for receiving log.

#### Parameters

- severity** Current log level
- message** Log message
- user\_data** User-defined data

Definition at line **1833** of file **ObTypes.h**.

#### ◆ **ob\_playback\_status\_changed\_callback**

```
typedef void(* ob_playback_status_changed_callback) (ob_playback_status status, void *user_data)
```

Definition at line **1835** of file **ObTypes.h**.

## Enumeration Type Documentation

#### ◆ **OBPermissionType**

enum **OBPermissionType**

the permission type of api or property

Enumerator

OB_PERMISSION_DENY	no permission
OB_PERMISSION_READ	can read
OB_PERMISSION_WRITE	can write
OB_PERMISSION_READ_WRITE	can read and write
OB_PERMISSION_ANY	any situation above

Definition at line **63** of file **ObTypes.h**.

◆ **OBStatus**

enum **OBStatus**

error code

Enumerator

OB_STATUS_OK	status ok
OB_STATUS_ERROR	status error

Definition at line **75** of file **ObTypes.h**.

◆ **OBLogSeverity**

enum **OBLogSeverity**

log level, the higher the level, the stronger the log filter

Enumerator

OB_LOG_SEVERITY_DEBUG	debug
OB_LOG_SEVERITY_INFO	information
OB_LOG_SEVERITY_WARN	warning
OB_LOG_SEVERITY_ERROR	error
OB_LOG_SEVERITY_FATAL	fatal error
OB_LOG_SEVERITY_OFF	off (close LOG)

Definition at line **84** of file **ObTypes.h**.

◆ OBExceptionType

enum **OBExceptionType**

The exception types in the SDK, through the exception type, you can easily determine the specific type of error. For detailed error API interface functions and error logs, please refer to the information of [ob\\_error](#).

Enumerator

OB_EXCEPTION_TYPE_UNKNOWN	Unknown error, an error not clearly defined by the SDK
OB_EXCEPTION_STD_EXCEPTION	< Standard exception, an error caused by the standard library Camera/Device has been disconnected, the camera/device is not available
OB_EXCEPTION_TYPE_CAMERA_DISCONNECTED	
OB_EXCEPTION_TYPE_PLATFORM	An error in the SDK adaptation platform layer, which means an error in the implementation of a specific system platform
OB_EXCEPTION_TYPE_INVALID_VALUE	Invalid parameter type exception, need to check input parameter
OB_EXCEPTION_TYPE_WRONG_API_CALL_SEQUENCE	Wrong API call sequence, the API is called in the wrong order or the wrong parameter is passed
OB_EXCEPTION_TYPE_NOT_IMPLEMENTED	SDK and firmware have not yet implemented this function or feature
OB_EXCEPTION_TYPE_IO	SDK access IO exception error
OB_EXCEPTION_TYPE_MEMORY	SDK access and use memory errors. For example, the frame fails to allocate memory
OB_EXCEPTION_TYPE_UNSUPPORTED_OPERATION	Unsupported operation type error by SDK or device

Definition at line [99](#) of file [ObTypes.h](#).

◆ **OBSensorType**

enum **OBSensorType**

Enumeration value describing the sensor type.

Enumerator

OB_SENSOR_UNKNOWN	Unknown type sensor
OB_SENSOR_IR	IR
OB_SENSOR_COLOR	Color
OB_SENSOR_DEPTH	Depth
OB_SENSOR_ACCEL	Accel
OB_SENSOR_GYRO	Gyro
OB_SENSOR_IR_LEFT	left IR for stereo camera
OB_SENSOR_IR_RIGHT	Right IR for stereo camera
OB_SENSOR_RAW_PHASE	Raw Phase
OB_SENSOR_CONFIDENCE	Confidence
OB_SENSOR_TYPE_COUNT	

Definition at line **128** of file **ObTypes.h**.

◆ **OBStreamType**

enum **OBStreamType**

Enumeration value describing the type of data stream.

Enumerator

OB_STREAM_UNKNOWN	Unknown type stream
OB_STREAM_VIDEO	Video stream (infrared, color, depth streams are all video streams)
OB_STREAM_IR	IR stream
OB_STREAM_COLOR	color stream
OB_STREAM_DEPTH	depth stream
OB_STREAM_ACCEL	Accelerometer data stream
OB_STREAM_GYRO	Gyroscope data stream
OB_STREAM_IR_LEFT	Left IR stream for stereo camera
OB_STREAM_IR_RIGHT	Right IR stream for stereo camera
OB_STREAM_RAW_PHASE	RawPhase Stream
OB_STREAM_CONFIDENCE	Confidence Stream
OB_STREAM_TYPE_COUNT	The total number of stream type, is not a valid stream type

Definition at line **146** of file **ObTypes.h**.

◆ **OBFrameType**

enum **OBFrameType**

Enumeration value describing the type of frame.

Enumerator

OB_FRAME_UNKNOWN	Unknown frame type
OB_FRAME_VIDEO	Video frame
OB_FRAME_IR	IR frame
OB_FRAME_COLOR	Color frame
OB_FRAME_DEPTH	Depth frame
OB_FRAME_ACCEL	Accelerometer data frame
OB_FRAME_SET	Frame collection (internally contains a variety of data frames)
OB_FRAME_POINTS	Point cloud frame
OB_FRAME_GYRO	Gyroscope data frame
OB_FRAME_IR_LEFT	Left IR frame for stereo camera
OB_FRAME_IR_RIGHT	Right IR frame for stereo camera
OB_FRAME_RAW_PHASE	Raw Phase frame
OB_FRAME_CONFIDENCE	Confidence frame
OB_FRAME_TYPE_COUNT	The total number of frame types, is not a valid frame type

Definition at line **165** of file **ObTypes.h**.

◆ **OBPixelType**

## enum **OBPixelType**

Enumeration value describing the pixel type of frame (usually used for depth frame)

Enumerator

- OB\_PIXEL\_UNKNOWN
- OB\_PIXEL\_DEPTH
- OB\_PIXEL\_DISPARITY
- OB\_PIXEL\_RAW\_PHASE
- OB\_PIXEL\_TOF\_DEPTH

Definition at line **187** of file **ObTypes.h**.

## ◆ OBFormat

### enum **OBFormat**

Enumeration value describing the pixel format.

Enumerator

OB_FORMAT_UNKNOWN	unknown format
OB_FORMAT_YUYV	YUYV format
OB_FORMAT_YUY2	YUY2 format (the actual format is the same as YUYV)
OB_FORMAT_UYVY	UYVY format
OB_FORMAT_NV12	NV12 format
OB_FORMAT_NV21	NV21 format
OB_FORMAT_MJPG	MJPEG encoding format
OB_FORMAT_H264	H.264 encoding format
OB_FORMAT_H265	H.265 encoding format
OB_FORMAT_Y16	Y16 format, 16-bit per pixel, single-channel
OB_FORMAT_Y8	Y8 format, 8-bit per pixel, single-channel
OB_FORMAT_Y10	Y10 format, 10-bit per pixel, single-channel(SDK will unpack into Y16 by default)
OB_FORMAT_Y11	Y11 format, 11-bit per pixel, single-channel (SDK will unpack into Y16 by default)
OB_FORMAT_Y12	Y12 format, 12-bit per pixel, single-channel(SDK will unpack into Y16 by default)

OB_FORMAT_GRAY	GRAY (the actual format is the same as YUYV)
OB_FORMAT_HEVC	HEVC encoding format (the actual format is the same as H265)
OB_FORMAT_I420	I420 format
OB_FORMAT_ACCEL	Acceleration data format
OB_FORMAT_GYRO	Gyroscope data format
OB_FORMAT_POINT	XYZ 3D coordinate point format, <b>OBPoint</b>
OB_FORMAT_RGB_POINT	XYZ 3D coordinate point format with RGB information, <b>OBCColorPoint</b>
OB_FORMAT_RLE	RLE pressure test format (SDK will be unpacked into Y16 by default)
OB_FORMAT_RGB	RGB format (actual RGB888)
OB_FORMAT_BGR	BGR format (actual BGR888)
OB_FORMAT_Y14	Y14 format, 14-bit per pixel, single-channel (SDK will unpack into Y16 by default)
OB_FORMAT_BGRA	BGRA format
OB_FORMAT_COMPRESSED	Compression format
OB_FORMAT_RVL	RVL pressure test format (SDK will be unpacked into Y16 by default)
OB_FORMAT_Z16	Is same as Y16
OB_FORMAT_YV12	Is same as Y12, using for right ir stream
OB_FORMAT_BA81	Is same as Y8, using for right ir stream
OB_FORMAT_RGBA	RGBA format
OB_FORMAT_BYR2	byr2 format
OB_FORMAT_RW16	RAW16 format
OB_FORMAT_Y12C4	Y12C4 format

Definition at line **199** of file **ObTypes.h**.

## ◆ OBUpgradeState

enum **OBUpgradeState**

Enumeration value describing the firmware upgrade status.

Enumerator

STAT_DONE_WITH_DUPLICATES	update completed, but some files were duplicated and ignored
STAT_VERIFY_SUCCESS	Image file verify success
STAT_FILE_TRANSFER	file transfer
STAT_DONE	update completed
STAT_IN_PROGRESS	upgrade in process
STAT_START	start the upgrade
STAT_VERIFY_IMAGE	Image file verification
ERR_VERIFY	Verification failed
ERR_PROGRAM	Program execution failed
ERR_ERASE	Flash parameter failed
ERR_FLASH_TYPE	Flash type error
ERR_IMAGE_SIZE	Image file size error
ERR_OTHER	other errors
ERR_DDR	DDR access error
ERR_TIMEOUT	timeout error
ERR_MISMATCH	Mismatch firmware error
ERR_UNSUPPORT_DEV	Unsupported device error
ERR_INVALID_COUNT	invalid firmware/preset count

Definition at line [253](#) of file **ObTypes.h**.

◆ **OBFileTranState**

## enum **OBFileTranState**

Enumeration value describing the file transfer status.

Enumerator

FILE_TRAN_STAT_TRANSFER	File transfer
FILE_TRAN_STAT_DONE	File transfer succeeded
FILE_TRAN_STAT_PREPAR	Preparing
FILE_TRAN_ERR_DDR	DDR access failed
FILE_TRAN_ERR_NOT_ENOUGH_SPACE	Insufficient target space error
FILE_TRAN_ERR_PATH_NOT_WRITABLE	Destination path is not writable
FILE_TRAN_ERR_MD5_ERROR	MD5 checksum error
FILE_TRAN_ERR_WRITE_FLASH_ERROR	Write flash error
FILE_TRAN_ERR_TIMEOUT	Timeout error

Definition at line **278** of file **ObTypes.h**.

## ◆ OBDataTranState

enum **OBDataTranState**

Enumeration value describing the data transfer status.

Enumerator

DATA_TRAN_STAT_VERIFY_DONE	data verify done
DATA_TRAN_STAT_STOPPED	data transfer stoped
DATA_TRAN_STAT_DONE	data transfer completed
DATA_TRAN_STAT VERIFYING	data verifying
DATA_TRAN_STAT_TRANSFERRING	data transferring
DATA_TRAN_ERR_BUSY	Transmission is busy
DATA_TRAN_ERR_UNSUPPORTED	Not supported
DATA_TRAN_ERR_TRAN_FAILED	Transfer failed
DATA_TRAN_ERR_VERIFY_FAILED	Test failed
DATA_TRAN_ERR_OTHER	Other errors

Definition at line [294](#) of file **ObTypes.h**.

◆ **OBCameraDistortionModel**

enum **OBCameraDistortionModel**

Distortion model: defines how pixel coordinates should be mapped to sensor coordinates.

Enumerator

OB_DISTORTION_NONE	Rectilinear images. No distortion compensation required.
OB_DISTORTION_MODIFIED_BROWN_CONRADY	Equivalent to Brown-Conrady distortion, except that tangential distortion is applied to radially distorted points
OB_DISTORTION_INVERSE_BROWN_CONRADY	Equivalent to Brown-Conrady distortion, except undistorts image instead of distorting it
OB_DISTORTION_BROWN_CONRADY	Unmodified Brown-Conrady distortion model
OB_DISTORTION_BROWN_CONRADY_K6	Unmodified Brown-Conrady distortion model with k6 supported
OB_DISTORTION_KANNALA_BRANDT4	Kannala-Brandt distortion model

Definition at line [374](#) of file **ObTypes.h**.

◆ **OBAlignMode**

enum **OBAlignMode**

Alignment mode.

Enumerator

ALIGN_DISABLE	Turn off alignment
ALIGN_D2C_HW_MODE	Hardware D2C alignment mode
ALIGN_D2C_SW_MODE	Software D2C alignment mode

Definition at line [506](#) of file **ObTypes.h**.

◆ **OBCameraPerformanceMode**

enum **OBCameraPerformanceMode**

Camera performance mode.

Enumerator

ADAPTIVE\_PERFORMANCE\_MODE Camera adaptive mode

HIGH\_PERFORMANCE\_MODE High Performance Mode

Definition at line **516** of file **ObTypes.h**.

◆ **OBConvertFormat**

enum **OBConvertFormat**

Enumeration of format conversion types.

Enumerator

FORMAT_YUYV_TO_RGB	YUYV to RGB
FORMAT_I420_TO_RGB	I420 to RGB
FORMAT_NV21_TO_RGB	NV21 to RGB
FORMAT_NV12_TO_RGB	NV12 to RGB
FORMAT_MJPG_TO_I420	MJPG to I420
FORMAT_RGB_TO_BGR	RGB888 to BGR
FORMAT_MJPG_TO_NV21	MJPG to NV21
FORMAT_MJPG_TO_RGB	MJPG to RGB
FORMAT_MJPG_TO_BGR	MJPG to BGR
FORMAT_MJPG_TO_BGRA	MJPG to BGRA
FORMAT_UYVY_TO_RGB	UYVY to RGB
FORMAT_BGR_TO_RGB	BGR to RGB
FORMAT_MJPG_TO_NV12	MJPG to NV12
FORMAT_YUYV_TO_BGR	YUYV to BGR
FORMAT_YUYV_TO_RGBA	YUYV to RGBA
FORMAT_YUYV_TO_BGRA	YUYV to BGRA
FORMAT_YUYV_TO_Y16	YUYV to Y16
FORMAT_YUYV_TO_Y8	YUYV to Y8
FORMAT_RGBA_TO_RGB	RGBA to RGB
FORMAT_BGRA_TO_BGR	BGRA to BGR
FORMAT_Y16_TO_RGB	Y16 to RGB
FORMAT_Y8_TO_RGB	Y8 to RGB

Definition at line [535](#) of file **ObTypes.h**.

◆ OBIMUSampleRate

enum **OBIMUSampleRate**

Enumeration of IMU sample rate values (gyroscope or accelerometer)

Enumerator

OB_SAMPLE_RATE_UNKNOWN	
OB_SAMPLE_RATE_1_5625_HZ	1.5625Hz
OB_SAMPLE_RATE_3_125_HZ	3.125Hz
OB_SAMPLE_RATE_6_25_HZ	6.25Hz
OB_SAMPLE_RATE_12_5_HZ	12.5Hz
OB_SAMPLE_RATE_25_HZ	25Hz
OB_SAMPLE_RATE_50_HZ	50Hz
OB_SAMPLE_RATE_100_HZ	100Hz
OB_SAMPLE_RATE_200_HZ	200Hz
OB_SAMPLE_RATE_500_HZ	500Hz
OB_SAMPLE_RATE_1_KHZ	1KHz
OB_SAMPLE_RATE_2_KHZ	2KHz
OB_SAMPLE_RATE_4_KHZ	4KHz
OB_SAMPLE_RATE_8_KHZ	8KHz
OB_SAMPLE_RATE_16_KHZ	16KHz
OB_SAMPLE_RATE_32_KHZ	32Hz
OB_SAMPLE_RATE_400_HZ	400Hz
OB_SAMPLE_RATE_800_HZ	800Hz

Definition at line [579](#) of file **ObTypes.h**.

◆ OBGyroFullScaleRange

enum **OBGyroFullScaleRange**

Enumeration of gyroscope ranges.

Enumerator

OB_GYRO_FS_UNKNOWN	
OB_GYRO_FS_16dps	16 degrees per second
OB_GYRO_FS_31dps	31 degrees per second
OB_GYRO_FS_62dps	62 degrees per second
OB_GYRO_FS_125dps	125 degrees per second
OB_GYRO_FS_250dps	250 degrees per second
OB_GYRO_FS_500dps	500 degrees per second
OB_GYRO_FS_1000dps	1000 degrees per second
OB_GYRO_FS_2000dps	2000 degrees per second
OB_GYRO_FS_400dps	400 degrees per second
OB_GYRO_FS_800dps	800 degrees per second

Definition at line **604** of file **ObTypes.h**.

◆ **OBAccelFullScaleRange**

## enum **OBAccelFullScaleRange**

Enumeration of accelerometer ranges.

Enumerator

OB_ACCEL_FS_UNKNOWN	
OB_ACCEL_FS_2g	1x the acceleration of gravity
OB_ACCEL_FS_4g	4x the acceleration of gravity
OB_ACCEL_FS_8g	8x the acceleration of gravity
OB_ACCEL_FS_16g	16x the acceleration of gravity
OB_ACCEL_FS_3g	3x the acceleration of gravity
OB_ACCEL_FS_6g	6x the acceleration of gravity
OB_ACCEL_FS_12g	12x the acceleration of gravity
OB_ACCEL_FS_24g	24x the acceleration of gravity

Definition at line **622** of file **ObTypes.h**.

## ◆ OBDepthCroppingMode

### enum **OBDepthCroppingMode**

Enumeration for depth crop modes.

Enumerator

DEPTH_CROPPING_MODE_AUTO	Automatic mode
DEPTH_CROPPING_MODE_CLOSE	Close crop
DEPTH_CROPPING_MODE_OPEN	Open crop

Definition at line **669** of file **ObTypes.h**.

## ◆ OBDeviceType

## enum **OBDeviceType**

Enumeration for device types.

Enumerator

OB_DEVICE_TYPE_UNKNOWN	Unknown device type
OB_STRUCTURED_LIGHT_MONOCULAR_CAMERA	Monocular structured light camera
OB_STRUCTURED_LIGHT_BINOCULAR_CAMERA	Binocular structured light camera
OB_TOF_CAMERA	Time-of-flight camera

Definition at line **679** of file **ObTypes.h**.

## ◆ **OBMediaType**

### enum **OBMediaType**

Enumeration for types of media to record or playback.

Enumerator

OB_MEDIA_COLOR_STREAM	Color stream
OB_MEDIA_DEPTH_STREAM	Depth stream
OB_MEDIA_IR_STREAM	Infrared stream
OB_MEDIA_GYRO_STREAM	Gyroscope stream
OB_MEDIA_ACCEL_STREAM	Accelerometer stream
OB_MEDIA_CAMERA_PARAM	Camera parameter
OB_MEDIA_DEVICE_INFO	Device information
OB_MEDIA_STREAM_INFO	Stream information
OB_MEDIA_IR_LEFT_STREAM	Left infrared stream
OB_MEDIA_IR_RIGHT_STREAM	Right infrared stream
OB_MEDIA_ALL	All media data types

Definition at line **690** of file **ObTypes.h**.

## ◆ **OBMediaState**

## enum **OBMediaState**

Enumeration for record playback status.

Enumerator

OB_MEDIA_BEGIN	Begin
OB_MEDIA_PAUSE	Pause
OB_MEDIA_RESUME	Resume
OB_MEDIA_END	End

Definition at line [710](#) of file **ObTypes.h**.

## ◆ OBDepthPrecisionLevel

### enum **OBDepthPrecisionLevel**

Enumeration for depth precision levels.

#### Attention

The depth precision level does not completely determine the depth unit and real precision, and the influence of the data packaging format needs to be considered. The specific unit can be obtained through `getValueScale()` of `DepthFrame`

Enumerator

OB_PRECISION_1MM	1mm
OB_PRECISION_0MM8	0.8mm
OB_PRECISION_0MM4	0.4mm
OB_PRECISION_0MM1	0.1mm
OB_PRECISION_0MM2	0.2mm
OB_PRECISION_0MM5	0.5mm
OB_PRECISION_0MM05	0.05mm
OB_PRECISION_UNKNOWN	
OB_PRECISION_COUNT	

Definition at line [723](#) of file **ObTypes.h**.

## ◆ OBTofFilterRange

## enum **OBTofFilterRange**

Enumeration for TOF filter scene ranges.

Enumerator

OB_TOF_FILTER_RANGE_CLOSE	Close range
OB_TOF_FILTER_RANGE_MIDDLE	Middle range
OB_TOF_FILTER_RANGE_LONG	Long range
OB_TOF_FILTER_RANGE_DEBUG	Debug range

Definition at line **758** of file **ObTypes.h**.

## ◆ OBCompressionMode

### enum **OBCompressionMode**

Compression mode.

Enumerator

OB_COMPRESSION LOSSLESS	Lossless compression mode
OB_COMPRESSION LOSSY	Lossy compression mode

Definition at line **804** of file **ObTypes.h**.

## ◆ OBSyncMode

### enum **OBSyncMode**

Sync mode.

#### **Deprecated**

This define is deprecated, please use **ob\_multi\_device\_sync\_mode** instead

Enumerator

OB_SYNC_MODE_CLOSE	Close synchronize mode.
	Single device, neither process input trigger signal nor output trigger signal
	Each Sensor in a single device automatically triggers
OB_SYNC_MODE_STANDALONE	Standalone synchronize mode.
	Single device, neither process input trigger signal nor output trigger signal
	Inside single device, RGB as Major sensor: RGB -> IR/Depth/TOF
OB_SYNC_MODE_PRIMARY	Primary synchronize mode.
	Primary device. Ignore process input trigger signal, only output trigger signal to secondary devices.
	Inside single device, RGB as Major sensor: RGB -> IR/Depth/TOF
OB_SYNC_MODE_SECONDARY	Secondary synchronize mode.
	Secondary device. Both process input trigger signal and output trigger signal to other devices.
	Different sensors in a single devices receive trigger signals respectively: ext trigger -> RGB && ext trigger -> IR/Depth/TOF

### Attention

With the current Gemini 2 device set to this mode, each Sensor receives the first external trigger signal after the stream is turned on and starts timing self-triggering at the set frame rate until the stream is turned off

OB_SYNC_MODE_PRIMARY MCU_TRIGGER	MCU Primary synchronize mode.  Primary device. Ignore process input trigger signal, only output trigger signal to secondary devices.
OB_SYNC_MODE_PRIMARY IR_TRIGGER	Inside device, MCU is the primary signal source: MCU -> RGB && MCU -> IR/Depth/TOF  IR Primary synchronize mode.
OB_SYNC_MODE_PRIMARY_SOFT_TRIGGER	Primary device. Ignore process input trigger signal, only output trigger signal to secondary devices.  Inside device, IR is the primary signal source: IR/Depth/TOF -> RGB  Software trigger synchronize mode.
OB_SYNC_MODE_SECONDARY_SOFT_TRIGGER	Host, triggered by software control (receive the upper computer command trigger), at the same time to the trunk output trigger signal  Different sensors in a single machine receive trigger signals respectively: soft trigger -> RGB && soft trigger -> IR/Depth/TOF
<b>Attention</b>	
Support product: Gemini2	
OB_SYNC_MODE_IR_IMU_SYNC	Software trigger synchronize mode as secondary device.  The slave receives the external trigger signal (the external trigger signal comes from the soft trigger host) and outputs the trigger signal to the external relay.
OB_SYNC_MODE_UNKNOWN	Different sensors in a single machine receive trigger signals respectively: ext trigger -> RGB && ext trigger -> IR/Depth/TOF  IR and IMU sync signal.  Unknown type.

Definition at line [832](#) of file **ObTypes.h**.

◆ **OBDepthWorkModeTag**

enum **OBDepthWorkModeTag**

Preset tag.

Enumerator

OB\_DEVICE\_DEPTH\_WORK\_MODE

OB\_CUSTOM\_DEPTH\_WORK\_MODE

Definition at line **969** of file **ObTypes.h**.

◆ **OBHoleFillingMode**

enum **OBHoleFillingMode**

Hole fillig mode.

Enumerator

OB\_HOLE\_FILL\_TOP

OB\_HOLE\_FILL\_NEAREST

OB\_HOLE\_FILL\_FAAREST

Definition at line **1006** of file **ObTypes.h**.

◆ **OB\_EDGE\_NOISE\_REMOVAL\_TYPE**

enum **OB\_EDGE\_NOISE\_REMOVAL\_TYPE**

Enumerator

OB\_MG\_FILTER

OB\_MGH\_FILTER

OB\_MGA\_FILTER

OB\_MGC\_FILTER

Definition at line **1020** of file **ObTypes.h**.

## ◆ OB\_DDO\_NOISE\_REMOVAL\_TYPE

enum **OB\_DDO\_NOISE\_REMOVAL\_TYPE**

Denoising method.

Enumerator

OB\_NR\_LUT

OB\_NR\_OVERALL

Definition at line **1039** of file **ObTypes.h**.

## ◆ OB\_CMD\_VERSION

enum **OB\_CMD\_VERSION**

Command version associated with property id.

Enumerator

OB\_CMD\_VERSION\_V0                    Version 1.0.

OB\_CMD\_VERSION\_V1                    Version 2.0.

OB\_CMD\_VERSION\_V2                    Version 3.0.

OB\_CMD\_VERSION\_V3                    Version 4.0.

OB\_CMD\_VERSION\_NOVERSION

OB\_CMD\_VERSION\_INVALID              Invalid version.

Definition at line **1074** of file **ObTypes.h**.

## ◆ OBCommunicationType

enum **OBCommunicationType**

Device communication mode.

Enumerator

OB\_COMM\_USB USB.

OB\_COMM\_NET Ethernet.

Definition at line [1118](#) of file **ObTypes.h**.

◆ **OBUSBPowerState**

enum **OBUSBPowerState**

USB power status.

Enumerator

OB\_USB\_POWER\_NO\_PLUGIN No plugin.

OB\_USB\_POWER\_5V\_0A9 5V/0.9A

OB\_USB\_POWER\_5V\_1A5 5V/1.5A

OB\_USB\_POWER\_5V\_3A0 5V/3.0A

Definition at line [1127](#) of file **ObTypes.h**.

◆ **OBDCPowerState**

enum **OBDCPowerState**

DC power status.

Enumerator

OB\_DC\_POWER\_NO\_PLUGIN No plugin.

OB\_DC\_POWER\_PLUGIN Plugin.

Definition at line [1138](#) of file **ObTypes.h**.

◆ **ob\_rotate\_degree\_type**

enum **ob\_rotate\_degree\_type**

Rotate degree.

Enumerator

OB_ROTATE_DEGREE_0	Rotate 0.
OB_ROTATE_DEGREE_90	Rotate 90.
OB_ROTATE_DEGREE_180	Rotate 180.
OB_ROTATE_DEGREE_270	Rotate 270.

Definition at line **1147** of file **ObTypes.h**.

◆ **ob\_power\_line\_freq\_mode**

enum **ob\_power\_line\_freq\_mode**

Power line frequency mode, for color camera anti-flicker configuration.

Enumerator

OB_POWER_LINE_FREQ_MODE_CLOSE	Close.
OB_POWER_LINE_FREQ_MODE_50HZ	50Hz
OB_POWER_LINE_FREQ_MODE_60HZ	60Hz

Definition at line **1158** of file **ObTypes.h**.

◆ **OB\_FRAME\_AGGREGATE\_OUTPUT\_MODE**

enum **OB\_FRAME\_AGGREGATE\_OUTPUT\_MODE**

Frame aggregate output mode.

Enumerator

OB_FRAME_AGGREGATE_OUTPUT_ALL_TYPE_FRAME_REQUIRE	Only FrameSet that contains all types of data frames will be output.
--	--

OB_FRAME_AGGREGATE_OUTPUT_COLOR_FRAME_REQUIRE	Color Frame Require output mode.
	Suitable for Color using H264, H265 and other inter-frame encoding format open stream
	<b>Attention</b>
	In this mode, the user may return null when getting a non-Color type data frame from the acquired FrameSet
OB_FRAME_AGGREGATE_OUTPUT_ANY_SITUATION	FrameSet for any case will be output.
	<b>Attention</b>
	In this mode, the user may return null when getting the specified type of data frame from the acquired FrameSet
OB_FRAME_AGGREGATE_OUTPUT_DISABLE	Disable Frame Aggreate.
	<b>Attention</b>
	In this mode, All types of data frames will output independently.

Definition at line **1168** of file **ObTypes.h**.

- ◆ OB\_COORDINATE\_SYSTEM\_TYPE

## enum **OB\_COORDINATE\_SYSTEM\_TYPE**

Enumeration of point cloud coordinate system types.

Enumerator

OB\_LEFT\_HAND\_COORDINATE\_SYSTEM

OB\_RIGHT\_HAND\_COORDINATE\_SYSTEM

Definition at line **1201** of file **ObTypes.h**.

## ◆ OB\_DEVICE DEVELOPMENT MODE

### enum **OB\_DEVICE DEVELOPMENT MODE**

Enumeration of device development modes.

Enumerator

OB\_USER\_MODE User mode (default mode), which provides full camera device functionality.

OB\_DEVELOPER\_MODE Developer mode, which allows developers to access the operating system and software/hardware resources on the device directly.

Definition at line **1210** of file **ObTypes.h**.

## ◆ ob\_multi\_device\_sync\_mode

### enum **ob\_multi\_device\_sync\_mode**

The synchronization mode of the device.

Enumerator

OB\_MULTI\_DEVICE\_SYNC\_MODE\_FREE\_RUN free run mode

The device does not synchronize with other devices,

The Color and Depth can be set to different frame rates.

OB_MULTI_DEVICE_SYNC_MODE_STANDALONE	standalone mode  The device does not synchronize with other devices.
OB_MULTI_DEVICE_SYNC_MODE_PRIMARY	The Color and Depth should be set to same frame rates, the Color and Depth will be synchronized.  primary mode  The device is the primary device in the multi-device system, it will output the trigger signal via VSYNC_OUT pin on synchronization port by default.
	The Color and Depth should be set to same frame rates, the Color and Depth will be synchronized and can be adjusted by colorDelayUs, depthDelayUs or trigger2ImageDelayUs.

OB_MULTI_DEVICE_SYNC_MODE_SECONDARY	secondary mode
	The device is the secondary device in the multi-device system, it will receive the trigger signal via VSYNC_IN pin on synchronization port. It will out the trigger signal via VSYNC_OUT pin on synchronization port by default.
	The Color and Depth should be set to same frame rates, the Color and Depth will be synchronized and can be adjusted by colorDelayUs, depthDelayUs or trigger2ImageDelayUs.
	After starting the stream, the device will wait for the trigger signal to start capturing images, and will stop capturing images when the trigger signal is stopped.

### **Attention**

The frequency of the trigger signal should be same as the frame rate of the stream profile which is set when starting the stream.

OB_MULTI_DEVICE_SYNC_MODE_SECONDARY_SYNCED	secondary synced mode
	The device is the secondary device in the multi-device system, it will receive the trigger signal via VSYNC_IN pin on synchronization port. It will out the trigger signal via VSYNC_OUT pin on synchronization port by default.
	The Color and Depth should be set to same frame rates, the Color and Depth will be synchronized and can be adjusted by colorDelayUs, depthDelayUs or trigger2ImageDelayUs.
	After starting the stream, the device will be immediately start capturing images, and will adjust the capture time when the trigger signal is received to synchronize with the primary device. If the trigger signal is stopped, the device will still capture images.

### **Attention**

The frequency of the trigger signal should be same as the frame rate of the stream profile which is set when starting the stream.

OB_MULTI_DEVICE_SYNC_MODE_SOFTWARE_TRIGGERING	software triggering mode	<p>The device will start one time image capture after receiving the capture command and will output the trigger signal via VSYNC_OUT pin by default. The capture command can be sent from host by call <a href="#"><b>ob_device_trigger_capture</b></a>. The number of images captured each time can be set by framesPerTrigger.</p> <p>The Color and Depth should be set to same frame rates, the Color and Depth will be synchronized and can be adjusted by colorDelayUs, depthDelayUs or trigger2ImageDelayUs.</p> <p>The frequency of the user call <a href="#"><b>ob_device_trigger_capture</b></a> to send the capture command multiplied by the number of frames per trigger should be less than the frame rate of the stream profile which is set when starting the stream.</p>
OB_MULTI_DEVICE_SYNC_MODE_HARDWARE_TRIGGERING	hardware triggering mode	<p>The device will start one time image capture after receiving the trigger signal via VSYNC_IN pin on synchronization port and will output the trigger signal via VSYNC_OUT pin by default. The number of images captured each time can be set by framesPerTrigger.</p> <p>The Color and Depth should be set to same frame rates, the</p>

Color and Depth will be synchronized and can be adjusted by colorDelayUs, depthDelayUs or trigger2ImageDelayUs.

### Attention

The frequency of the trigger signal multiplied by the number of frames per trigger should be less than the frame rate of the stream profile which is set when starting the stream.

The trigger signal input via VSYNC\_IN pin on synchronization port should be output by other device via VSYNC\_OUT pin in hardware triggering mode or software triggering mode.

Due to different models may have different signal input requirements, please do not use different models to output trigger signal as input-trigger signal.

OB\_MULTI\_DEVICE\_SYNC\_MODE\_IR\_IMU\_SYNC

IR and IMU sync mode.

Definition at line [1226](#) of file [ObTypes.h](#).

- ◆ OBFilterConfigValueType

## enum **OBFilterConfigValueType**

Enumerator

OB\_FILTER\_CONFIG\_VALUE\_TYPE\_INVALID

OB\_FILTER\_CONFIG\_VALUE\_TYPE\_INT

OB\_FILTER\_CONFIG\_VALUE\_TYPE\_FLOAT

OB\_FILTER\_CONFIG\_VALUE\_TYPE\_BOOLEAN

Definition at line **1449** of file **ObTypes.h**.

## ◆ **ob\_frame\_metadata\_type**

### enum **ob\_frame\_metadata\_type**

Frame metadata types.

The frame metadata is a set of meta info generated by the device for current individual frame.

Enumerator

OB\_FRAME\_METADATA\_TYPE\_TIMESTAMP

Timestamp when the frame is captured.

### **Attention**

Different device models may have different units. It is recommended to use the timestamp related functions to get the timestamp in the correct units.

OB_FRAME_METADATA_TYPE_SENSOR_TIMESTAMP	Timestamp in the middle of the capture.
	Usually is the middle of the exposure time.
	<b>Attention</b>
	Different device models may have different units.
OB_FRAME_METADATA_TYPE_FRAME_NUMBER	The number of current frame.
OB_FRAME_METADATA_TYPE_AUTO_EXPOSURE	Auto exposure status.
	If the value is 0, it means the auto exposure is disabled.
	Otherwise, it means the auto exposure is enabled.
OB_FRAME_METADATA_TYPE_EXPOSURE	Exposure time.
	<b>Attention</b>
	Different sensor may have different units. Usually, it is 100us for color sensor and 1us for depth/infrared sensor.
OB_FRAME_METADATA_TYPE_GAIN	Gain.
	<b>Attention</b>
	For some device models, the gain value represents the gain level, not the multiplier.

OB_FRAME_METADATA_TYPE_AUTO_WHITE_BALANCE	Auto white balance status.
	If the value is 0, it means the auto white balance is disabled. Otherwise, it means the auto white balance is enabled.
OB_FRAME_METADATA_TYPE_WHITE_BALANCE	White balance.
OB_FRAME_METADATA_TYPE_BRIGHTNESS	Brightness.
OB_FRAME_METADATA_TYPE_CONTRAST	Contrast.
OB_FRAME_METADATA_TYPE_SATURATION	Saturation.
OB_FRAME_METADATA_TYPE_SHARPNESS	Sharpness.
OB_FRAME_METADATA_TYPE_BACKLIGHT_COMPENSATION	Backlight compensation.
OB_FRAME_METADATA_TYPE_HUE	Hue.
OB_FRAME_METADATA_TYPE_GAMMA	Gamma.
OB_FRAME_METADATA_TYPE_POWER_LINE_FREQUENCY	Power line frequency.
OB_FRAME_METADATA_TYPE_LOW_LIGHT_COMPENSATION	For anti-flickering, 0: Close, 1: 50Hz, 2: 60Hz, 3: Auto Low light compensation.
<b>Attention</b>	
	The low light compensation is a feature inside the device, and can not manually control it.
OB_FRAME_METADATA_TYPE_MANUAL_WHITE_BALANCE	Manual white balance setting.
OB_FRAME_METADATA_TYPE_ACTUAL_FRAME_RATE	Actual frame rate.
	The actual frame rate will be calculated according to the exposure time and other parameters.
OB_FRAME_METADATA_TYPE_FRAME_RATE	Frame rate.
OB_FRAME_METADATA_TYPE_AE_ROI_LEFT	Left region of interest for the auto exposure Algorithm.
OB_FRAME_METADATA_TYPE_AE_ROI_TOP	Top region of interest for the auto exposure Algorithm.

OB_FRAME_METADATA_TYPE_AE_ROI_RIGHT	Right region of interest for the auto exposure Algorithm.
OB_FRAME_METADATA_TYPE_AE_ROI_BOTTOM	Bottom region of interest for the auto exposure Algorithm.
OB_FRAME_METADATA_TYPE_EXPOSURE_PRIORITY	Exposure priority.
OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_NAME	HDR sequence name.
OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_SIZE	HDR sequence size.
OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_INDEX	HDR sequence index.
OB_FRAME_METADATA_TYPE_LASER_POWER	Laser power value in mW.
<b>Attention</b>	
The laser power value is an approximate estimation.	
OB_FRAME_METADATA_TYPE_LASER_POWER_LEVEL	Laser power level.
OB_FRAME_METADATA_TYPE_LASER_STATUS	Laser status.
OB_FRAME_METADATA_TYPE_GPIO_INPUT_DATA	0: Laser off, 1: Laser on
OB_FRAME_METADATA_TYPE_DISPARITY_SEARCH_OFFSET	GPIO input data.
OB_FRAME_METADATA_TYPE_DISPARITY_SEARCH_RANGE	disparity search offset value
OB_FRAME_METADATA_TYPE_COUNT	disparity search range
The number of frame metadata types, using for types iterating.	
<b>Attention</b>	
It is not a valid frame metadata type	

Definition at line [1490](#) of file [ObTypes.h](#).

- ◆ [ob\\_uvc\\_backend\\_type](#)

## enum **ob\_uvc\_backend\_type**

For Linux, there are two ways to access the UVC device, libuvc and v4l2. The backend type is used to select the backend to access the device.

Enumerator

OB_UVC_BACKEND_TYPE_AUTO	Auto detect system capabilities and device hint to select backend.
OB_UVC_BACKEND_TYPE_LIBUVC	Use libuvc backend to access the UVC device.
OB_UVC_BACKEND_TYPE_V4L2	Use v4l2 backend to access the UVC device.
OB_UVC_BACKEND_TYPE_MSFM	Use MSFM backend to access the UVC device.

Definition at line **1691** of file **ObTypes.h**.

## ◆ **ob\_playback\_status**

### enum **ob\_playback\_status**

The playback status of the media.

Enumerator

OB_PLAYBACK_UNKNOWN	
OB_PLAYBACK_PLAYING	The media is playing
OB_PLAYBACK_PAUSED	The media is paused
OB_PLAYBACK_STOPPED	The media is stopped
OB_PLAYBACK_COUNT	

Definition at line **1720** of file **ObTypes.h**.

# ObTypes.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8
9 #pragma once
10
11 #include "Export.h"
12
13 #include <stdbool.h>
14 #include <stdint.h>
15
16 #pragma pack(push, 1) // struct 1-byte align
17
18 #ifdef __cplusplus
19 extern "C" {
20 #endif
21
22 typedef struct ob_context_t          ob_context;
23 typedef struct ob_device_t           ob_device;
24 typedef struct ob_device_info_t      ob_device_info;
25 typedef struct ob_device_list_t      ob_device_list;
26 typedef struct ob_record_device_t    ob_record_device;
27 typedef struct ob_playback_device_t  ob_playback_device;
28 typedef struct ob_camera_param_list_t ob_camera_param_list;
29 typedef struct ob_sensor_t          ob_sensor;
30 typedef struct ob_sensor_list_t      ob_sensor_list;
31 typedef struct ob_stream_profile_t   ob_stream_profile;
32 typedef struct ob_stream_profile_list_t ob_stream_profile_list;
33 typedef struct ob_frame_t           ob_frame;
34 typedef struct ob_filter_t          ob_filter;
35 typedef struct ob_filter_list_t      ob_filter_list;
36 typedef struct ob_pipeline_t         ob_pipeline;
37 typedef struct ob_config_t          ob_config;
38 typedef struct ob_depth_work_mode_list_t ob_depth_work_mode_list;
39 typedef struct ob_device_preset_list_t ob_device_preset_list;
40 typedef struct ob_filter_config_schema_list_t ob_filter_config_schema_list;
41 typedef struct ob_device_frame_interleave_list_t ob_device_frame_interleave_list;
42 typedef struct ob_preset_resolution_config_list_t ob_preset_resolution_config_list;
43
44 #define OB_WIDTH_ANY 0
45 #define OB_HEIGHT_ANY 0
46 #define OB_FPS_ANY 0
47 #define OB_FORMAT_ANY OB_FORMAT_UNKNOWN
48 #define OB_PROFILE_DEFAULT 0
49 #define OB_DEFAULT_STRIDE_BYTES 0
50 #define OB_ACCEL_FULL_SCALE_RANGE_ANY OB_ACCEL_FS_UNKNOWN
51 #define OB_ACCEL_SAMPLE_RATE_ANY OB_SAMPLE_RATE_UNKNOWN
52 #define OB_GYRO_FULL_SCALE_RANGE_ANY OB_GYRO_FS_UNKNOWN
53 #define OB_GYRO_SAMPLE_RATE_ANY OB_SAMPLE_RATE_UNKNOWN
54
58 #define OB_PATH_MAX (1024)
59
63 typedef enum {
64     OB_PERMISSION_DENY    = 0,
65     OB_PERMISSION_READ    = 1,
66     OB_PERMISSION_WRITE   = 2,
67     OB_PERMISSION_READ_WRITE = 3,
68     OB_PERMISSION_ANY     = 255,
```

```

69 } OBPermissionType,
70 ob_permission_type;
71
75 typedef enum {
76     OB_STATUS_OK    =0,
77     OB_STATUS_ERROR=1,
78 } OBStatus,
79 ob_status;
80
84 typedef enum {
85     OB_LOG_SEVERITY_DEBUG,
86     OB_LOG_SEVERITY_INFO,
87     OB_LOG_SEVERITY_WARN,
88     OB_LOG_SEVERITY_ERROR,
89     OB_LOG_SEVERITY_FATAL,
90     OB_LOG_SEVERITY_OFF
91 } OBLogSeverity,
92     ob_log_severity, DEVICE_LOG_SEVERITY_LEVEL, OBDeviceLogSeverityLevel, ob_device_log_severit
93     y_level;
94 #define OB_LOG_SEVERITY_NONE OB_LOG_SEVERITY_OFF
95
99 typedef enum {
100     OB_EXCEPTION_TYPE_UNKNOWN,
101     OB_EXCEPTION_STD_EXCEPTION,
102     OB_EXCEPTION_TYPE_CAMERA_DISCONNECTED,
103     OB_EXCEPTION_TYPE_PLATFORM,
105     OB_EXCEPTION_TYPE_INVALID_VALUE,
106     OB_EXCEPTION_TYPE_WRONG_API_CALL_SEQUENCE,
107     OB_EXCEPTION_TYPE_NOT_IMPLEMENTED,
108     OB_EXCEPTION_TYPE_IO,
109     OB_EXCEPTION_TYPE_MEMORY,
110     OB_EXCEPTION_TYPE_UNSUPPORTED_OPERATION,
111 } OBExceptionType,
112     ob_exception_type;
113
117 typedef struct ob_error {
118     ob_status      status;
119     char          message[256];
120     char          function[256];
121     char          args[256];
122     ob_exception_type exception_type;
123 } ob_error;
124
128 typedef enum {
129     OB_SENSOR_UNKNOWN   =0,
130     OB_SENSOR_IR        =1,
131     OB_SENSOR_COLOR     =2,
132     OB_SENSOR_DEPTH     =3,
133     OB_SENSOR_ACCEL     =4,
134     OB_SENSOR_GYRO      =5,
135     OB_SENSOR_IR_LEFT   =6,
136     OB_SENSOR_IR_RIGHT  =7,
137     OB_SENSOR_RAW_PHASE =8,
138     OB_SENSOR_CONFIDENCE=9,
139     OB_SENSOR_TYPE_COUNT,
140 } OBSDensorType,
141     ob_sensor_type;
142
146 typedef enum {
147     OB_STREAM_UNKNOWN   =-1,
148     OB_STREAM_VIDEO     =0,
149     OB_STREAM_IR        =1,
150     OB_STREAM_COLOR     =2,
151     OB_STREAM_DEPTH     =3,
152     OB_STREAM_ACCEL     =4,

```

```

153     OB_STREAM_GYRO = 5,
154     OB_STREAM_IR_LEFT = 6,
155     OB_STREAM_IR_RIGHT = 7,
156     OB_STREAM_RAW_PHASE=8,
157     OB_STREAM_CONFIDENCE=9,
158     OB_STREAM_TYPE_COUNT,
159 } OBStreamType,
160 ob_stream_type;
161
165 typedef enum {
166     OB_FRAME_UNKNOWN = -1,
167     OB_FRAME_VIDEO = 0,
168     OB_FRAME_IR = 1,
169     OB_FRAME_COLOR = 2,
170     OB_FRAME_DEPTH = 3,
171     OB_FRAME_ACCEL = 4,
172     OB_FRAME_SET = 5,
173     OB_FRAME_POINTS = 6,
174     OB_FRAME_GYRO = 7,
175     OB_FRAME_IR_LEFT = 8,
176     OB_FRAME_IR_RIGHT = 9,
177     OB_FRAME_RAW_PHASE=10,
178     OB_FRAME_CONFIDENCE=11,
179     OB_FRAME_TYPE_COUNT,
180 } OBFrameType,
181 ob_frame_type;
182
187 typedef enum {
188     OB_PIXEL_UNKNOWN = -1, // Unknown pixel type, or undefined pixel type for current frame
189     OB_PIXEL_DEPTH = 0, // Depth pixel type, the value of the pixel is the distance from the camera to the object
190     OB_PIXEL_DISPARITY=2, // Disparity for structured light camera
191     OB_PIXEL_RAW_PHASE=3, // Raw phase for tof camera
192     OB_PIXEL_TOF_DEPTH=4, // Depth for tof camera
193 } OBPixelType,
194 ob_pixel_type;
195
199 typedef enum {
200     OB_FORMAT_UNKNOWN = -1,
201     OB_FORMAT_YUYV = 0,
202     OB_FORMAT_YUY2 = 1,
203     OB_FORMAT_UYVY = 2,
204     OB_FORMAT_NV12 = 3,
205     OB_FORMAT_NV21 = 4,
206     OB_FORMAT_MJPG = 5,
207     OB_FORMAT_H264 = 6,
208     OB_FORMAT_H265 = 7,
209     OB_FORMAT_Y16 = 8,
210     OB_FORMAT_Y8 = 9,
211     OB_FORMAT_Y10 = 10,
212     OB_FORMAT_Y11 = 11,
213     OB_FORMAT_Y12 = 12,
214     OB_FORMAT_GRAY = 13,
215     OB_FORMAT_HEVC = 14,
216     OB_FORMAT_I420 = 15,
217     OB_FORMAT_ACCEL = 16,
218     OB_FORMAT_GYRO = 17,
219     OB_FORMAT_POINT = 19,
220     OB_FORMAT_RGB_POINT = 20,
221     OB_FORMAT_RLE = 21,
222     OB_FORMAT_RGB = 22,
223     OB_FORMAT_BGR = 23,
224     OB_FORMAT_Y14 = 24,
225     OB_FORMAT_BGRA = 25,
226     OB_FORMAT_COMPRESSED=26,
227     OB_FORMAT_PVR = 27

```

```

221     OB_FORMAT_YUV      = 27,
222     OB_FORMAT_Z16       = 28,
223     OB_FORMAT_YV12      = 29,
224     OB_FORMAT_BA81      = 30,
225     OB_FORMAT_RGBA       = 31,
226     OB_FORMAT_BYR2       = 32,
227     OB_FORMAT_RW16       = 33,
228     OB_FORMAT_Y12C4      = 34,
229 }
230 } OBFormat,
231 ob_format;
232
233
234
235
236
237
238 #define OB_FORMAT_RGB888 OB_FORMAT_RGB // Alias of OB_FORMAT_RGB for compatibility
239 #define OB_FORMAT_MJPEG OB_FORMAT_MJPG // Alias of OB_FORMAT_MJPG for compatibility
240
241 // Check if the format is a fixed data size format
242 #define IS_FIXED_SIZE_FORMAT(format) \
243     (format != OB_FORMAT_MJPG && format != OB_FORMAT_H264 && format != OB_FORMAT_H265 \
244         && format != OB_FORMAT_HEVC && format != OB_FORMAT_RLE \
245         && format != OB_FORMAT_RVL)
246
247 // Check if the format is a packed format, which means the data of pixels is not continuous or bytes aligned
248 // in memory
249 #define IS_PACKED_FORMAT(format) \
250     (format == OB_FORMAT_Y10 || format == OB_FORMAT_Y11 || format == OB_FORMAT_Y12 || format == \
251     OB_FORMAT_Y14 || format == OB_FORMAT_RLE)
252
253 typedef enum {
254     STAT_DONE_WITH_DUPLICATES = 6,
255     STAT_VERIFY_SUCCESS      = 5,
256     STAT_FILE_TRANSFER       = 4,
257     STAT_DONE                = 3,
258     STAT_IN_PROGRESS         = 2,
259     STAT_START               = 1,
260     STAT_VERIFY_IMAGE        = 0,
261     ERR_VERIFY               = -1,
262     ERR_PROGRAM              = -2,
263     ERR_ERASE                = -3,
264     ERR_FLASH_TYPE           = -4,
265     ERR_IMAGE_SIZE            = -5,
266     ERR_OTHER                 = -6,
267     ERR_DDR                   = -7,
268     ERR_TIMEOUT               = -8,
269     ERR_MISMATCH              = -9,
270     ERR_UNSUPORT_DEV          = -10,
271     ERR_INVALID_COUNT         = -11,
272 } OBUpgradeState,
273 OBFwUpdateState, ob_upgrade_state, ob_fw_update_state;
274
275
276
277
278 typedef enum {
279     FILE_TRAN_STAT_TRANSFER   = 2,
280     FILE_TRAN_STAT_DONE       = 1,
281     FILE_TRAN_STAT_PREPAR    = 0,
282     FILE_TRAN_ERR_DDR         = -1,
283     FILE_TRAN_ERR_NOT_ENOUGH_SPACE = -2,
284     FILE_TRAN_ERR_PATH_NOT_WRITABLE = -3,
285     FILE_TRAN_ERR_MD5_ERROR  = -4,
286     FILE_TRAN_ERR_WRITE_FLASH_ERROR = -5,
287     FILE_TRAN_ERR_TIMEOUT     = -6
288 } OBFfileTranState,
289 ob_file_tran_state;
290
291
292
293
294 typedef enum {
295     DATA_TRAN_STAT_VERIFY_DONE = 4,
296     DATA_TRAN_STAT_STOPPED    = 3,
297     DATA_TRAN_STAT_DONE        = 2,
298     DATA_TRAN_STAT VERIFYING = 1.

```

```

299     DATA_TRAN_STAT_TRANSFERRING=0,
300     DATA_TRAN_ERR_BUSY      =-1,
301     DATA_TRAN_ERR_UNSUPPORTED=-2,
302     DATA_TRAN_ERR_TRAN_FAILED=-3,
303     DATA_TRAN_ERR_VERIFY_FAILED=-4,
304     DATA_TRAN_ERR_OTHER     =-5
305 } OBDATAtranState,
306 ob_data_tran_state;
307
311 typedef struct {
312     uint8_t *data;
313     uint32_t size;
314     uint32_t offset;
315     uint32_t fullDataSize;
316 } OBDataChunk, ob_data_chunk;
317
321 typedef struct {
322     int32_t cur;
323     int32_t max;
324     int32_t min;
325     int32_t step;
326     int32_t def;
327 } OBIntPropertyRange, ob_int_property_range;
328
332 typedef struct {
333     float cur;
334     float max;
335     float min;
336     float step;
337     float def;
338 } OBFLOATPROPERTYRANGE, ob_float_property_range;
339
343 typedef struct {
344     uint16_t cur;
345     uint16_t max;
346     uint16_t min;
347     uint16_t step;
348     uint16_t def;
349 } OBUINT16PROPERTYRANGE, ob_uint16_property_range;
350
354 typedef struct {
355     uint8_t cur;
356     uint8_t max;
357     uint8_t min;
358     uint8_t step;
359     uint8_t def;
360 } OBUINT8PROPERTYRANGE, ob_uint8_property_range;
361
365 typedef struct {
366     bool cur;
367     bool max;
368     bool min;
369     bool step;
370     bool def;
371 } OBBBOOLPROPERTYRANGE, ob_bool_property_range;
372
374 typedef enum {
375     OB_DISTORTION_NONE,
376     OB_DISTORTION_MODIFIED_BROWN_CONRADY,
377     OB_DISTORTION_INVERSE_BROWN_CONRADY,
378     OB_DISTORTION_BROWN_CONRADY,
379     OB_DISTORTION_BROWN_CONRADY_K6,
380     OB_DISTORTION_BROWN_CONRADY_K7,
381     OB_DISTORTION_KANNALA_BRANDT4,
382 } OBCameraDistortionModel,
383 ob_camera_distortion_model;

```

```

384
385     typedef struct {
386         float fx;
387         float fy;
388         float cx;
389         float cy;
390         int16_t width;
391         int16_t height;
392     } OBCameraIntrinsic, ob_camera_intrinsic;
393
394
395
396     typedef struct {
397         double noiseDensity;
398         double randomWalk;
399         double referenceTemp;
400         double bias[3];
401         double gravity[3];
402         double scaleMisalignment[9];
403         double tempSlope[9];
404     } OBAccelIntrinsic, ob_accel_intrinsic;
405
406
407
408
409     typedef struct {
410         double noiseDensity;
411         double randomWalk;
412         double referenceTemp;
413         double bias[3];
414         double scaleMisalignment[9];
415         double tempSlope[9];
416     } OBGyroIntrinsic, ob_gyro_intrinsic;
417
418
419
420
421     typedef struct {
422         float k1;
423         float k2;
424         float k3;
425         float k4;
426         float k5;
427         float k6;
428         float p1;
429         float p2;
430         OBCameraDistortionModel model;
431     } OBCameraDistortion, ob_camera_distortion;
432
433
434
435
436
437     typedef struct {
438         float rot[9];
439         float trans[3];
440     } OBD2CTransform, ob_d2c_transform, OBTransform, ob_transform, OBExtrinsic, ob_extrinsic;
441
442
443
444
445     typedef struct {
446         OBCameraIntrinsic depthIntrinsic;
447         OBCameraIntrinsic rgbIntrinsic;
448         OBCameraDistortion depthDistortion;
449         OBCameraDistortion rgbDistortion;
450         OBD2CTransform transform;
451         bool isMirrored;
452     } OBCameraParam, ob_camera_param;
453
454
455
456
457     typedef struct {
458         int16_t width;
459         int16_t height;
460         int irDecimationFactor;
461         int depthDecimationFactor;
462     } OBPresetResolutionConfig, ob_preset_resolution_ratio_config;
463
464
465
466     typedef struct {
467         OBCameraIntrinsic intrinsics[OB_SENSOR_TYPE_COUNT];
468         OBCameraDistortion distortion[OB_SENSOR_TYPE_COUNT];
469         . . .

```

```

470     OBExtrinsic  extrinsics[OB_SENSOR_TYPE_COUNT]
471         [OB_SENSOR_TYPE_COUNT];
472 } OBCalibrationParam ob_calibration_param;
473
474 typedef struct {
475     int    margin_x_th;
476     int    margin_y_th;
477     int    limit_x_th;
478     int    limit_y_th;
479     uint32_t width;
480     uint32_t height;
481     bool   enable_direction;
482 } ob_margin_filter_config, OBMarginFilterConfig;
483
484
485 typedef struct {
486     uint32_t width;
487     uint32_t height;
488     int    max_width_left;
489     int    max_width_right;
490     int    max_radius;
491     int    margin_x_th;
492     int    margin_y_th;
493     int    limit_x_th;
494     int    limit_y_th;
495 } OBMGCFilterConfig, ob_mgc_filter_config;
496
497
498 typedef enum {
499     ALIGN_DISABLE,
500     ALIGN_D2C_HW_MODE,
501     ALIGN_D2C_SW_MODE,
502 } OBAccelMode,
503     ob_align_mode;
504
505
506     typedef enum {
507         ADAPTIVE_PERFORMANCE_MODE,
508         HIGH_PERFORMANCE_MODE,
509     } OBCameraPerformanceMode,
510         ob_camera_performance_mode;
511
512
513     typedef struct {
514         uint32_t x;
515         uint32_t y;
516         uint32_t width;
517         uint32_t height;
518     } OBRect, ob_rect;
519
520
521     typedef enum {
522         FORMAT_YUYV_TO_RGB=0,
523         FORMAT_I420_TO_RGB,
524         FORMAT_NV21_TO_RGB,
525         FORMAT_NV12_TO_RGB,
526         FORMAT_MJPG_TO_I420,
527         FORMAT_RGB_TO_BGR,
528         FORMAT_MJPG_TO_NV21,
529         FORMAT_MJPG_TO_RGB,
530         FORMAT_MJPG_TO_BGR,
531         FORMAT_MJPG_TO_BGRA,
532         FORMAT_UYVY_TO_RGB,
533         FORMAT_BGR_TO_RGB,
534         FORMAT_MJPG_TO_NV12,
535         FORMAT_YUYV_TO_BGR,
536         FORMAT_YUYV_TO_RGBA,
537         FORMAT_YUYV_TO_BGRA,
538         FORMAT_YUYV_TO_Y16,
539         FORMAT_YUYV_TO_Y8,
540         FORMAT_DCPA_TO_DCP
541
542
543
544
545
546
547
548
549
550
551
552
553
554

```

```

554     FORMAT_RGBA_TO_RGB,
555     FORMAT_BGRA_TO_BGR,
556     FORMAT_Y16_TO_RGB,
557     FORMAT_Y8_TO_RGB,
558 } OBConvertFormat,
559     ob_convert_format;
560
561 // DEPRECATED: Only used for old version program compatibility, will be completely deleted in subsequent iterative versions
562 #define FORMAT_MJPEG_TO_I420 FORMAT_MJPG_TO_I420
563 #define FORMAT_MJPEG_TO_NV21 FORMAT_MJPG_TO_NV21
564 #define FORMAT_MJPEG_TO_BGRA FORMAT_MJPG_TO_BGRA
565 #define FORMAT_YUYV_TO_RGB888 FORMAT_YUV_TO_RGB
566 #define FORMAT_I420_TO_RGB888 FORMAT_I420_TO_RGB
567 #define FORMAT_NV21_TO_RGB888 FORMAT_NV21_TO_RGB
568 #define FORMAT_NV12_TO_RGB888 FORMAT_NV12_TO_RGB
569 #define FORMAT_UYVY_TO_RGB888 FORMAT_UYVY_TO_RGB
570 #define FORMAT_MJPG_TO_RGB888 FORMAT_MJPG_TO_RGB
571 #define FORMAT_MJPG_TO_BGR888 FORMAT_MJPG_TO_BGR
572 #define FORMAT_MJPEG_TO_BGR888 FORMAT_MJPG_TO_BGR
573 #define FORMAT_RGB888_TO_BGR FORMAT_RGB_TO_BGR
574
575 typedef enum {
576     OB_SAMPLE_RATE_UNKNOWN = 0,
577     OB_SAMPLE_RATE_1_5625_HZ = 1,
578     OB_SAMPLE_RATE_3_125_HZ = 2,
579     OB_SAMPLE_RATE_6_25_HZ = 3,
580     OB_SAMPLE_RATE_12_5_HZ = 4,
581     OB_SAMPLE_RATE_25_HZ = 5,
582     OB_SAMPLE_RATE_50_HZ = 6,
583     OB_SAMPLE_RATE_100_HZ = 7,
584     OB_SAMPLE_RATE_200_HZ = 8,
585     OB_SAMPLE_RATE_500_HZ = 9,
586     OB_SAMPLE_RATE_1_KHZ = 10,
587     OB_SAMPLE_RATE_2_KHZ = 11,
588     OB_SAMPLE_RATE_4_KHZ = 12,
589     OB_SAMPLE_RATE_8_KHZ = 13,
590     OB_SAMPLE_RATE_16_KHZ = 14,
591     OB_SAMPLE_RATE_32_KHZ = 15,
592     OB_SAMPLE_RATE_400_HZ = 16,
593     OB_SAMPLE_RATE_800_HZ = 17,
594 } OBIMUSampleRate,
595     OBGyroSampleRate, ob_gyro_sample_rate, OBAccelSampleRate, ob_accel_sample_rate, OB_SAMPLE_RATE;
596
597 typedef enum {
598     OB_GYRO_FS_UNKNOWN = -1,
599     OB_GYRO_FS_16dps = 1,
600     OB_GYRO_FS_31dps = 2,
601     OB_GYRO_FS_62dps = 3,
602     OB_GYRO_FS_125dps = 4,
603     OB_GYRO_FS_250dps = 5,
604     OB_GYRO_FS_500dps = 6,
605     OB_GYRO_FS_1000dps = 7,
606     OB_GYRO_FS_2000dps = 8,
607     OB_GYRO_FS_400dps = 9,
608     OB_GYRO_FS_800dps = 10,
609 } OBGyroFullScaleRange,
610     ob_gyro_full_scale_range, OB_GYRO_FULL_SCALE_RANGE;
611
612 typedef enum {
613     OB_ACCEL_FS_UNKNOWN = -1,
614     OB_ACCEL_FS_2g = 1,
615     OB_ACCEL_FS_4g = 2,
616     OB_ACCEL_FS_8g = 3.

```

```

627     OB_ACCEL_FS_16g = 4,
628     OB_ACCEL_FS_3g = 5,
629     OB_ACCEL_FS_6g = 6,
630     OB_ACCEL_FS_12g = 7,
631     OB_ACCEL_FS_24g = 8,
632 } OBAccelFullScaleRange,
633     ob_accel_full_scale_range, OB_ACCEL_FULL_SCALE_RANGE;
634
635 typedef struct {
636     float x;
637     float y;
638     float z;
639 } OBAccelValue, OBGyroValue, OBFloat3D, ob_accel_value, ob_gyro_value, ob_float_3d;
640
641 typedef uint64_t OBDeviceState, ob_device_state;
642
643 typedef struct {
644     float cpuTemp;
645     float irTemp;
646     float ldmTemp;
647     float mainBoardTemp;
648     float tecTemp;
649     float imuTemp;
650     float rgbTemp;
651     float irLeftTemp;
652     float irRightTemp;
653     float chipTopTemp;
654     float chipBottomTemp;
655 } OBDeviceTemperature, ob_device_temperature, DEVICE_TEMPERATURE;
656
657 typedef enum {
658     DEPTH_CROPPING_MODE_AUTO = 0,
659     DEPTH_CROPPING_MODE_CLOSE = 1,
660     DEPTH_CROPPING_MODE_OPEN = 2,
661 } OBDepthCroppingMode,
662     ob_depth_cropping_mode, OB_DEPTH_CROPPING_MODE;
663
664 typedef enum {
665     OB_DEVICE_TYPE_UNKNOWN = -1,
666     OB_STRUCTURED_LIGHT_MONOCULAR_CAMERA = 0,
667     OB_STRUCTURED_LIGHT_BINOCULAR_CAMERA = 1,
668     OB_TOF_CAMERA = 2,
669 } OBDeviceType,
670     ob_device_type, OB_DEVICE_TYPE;
671
672 typedef enum {
673     OB_MEDIA_COLOR_STREAM = 1,
674     OB_MEDIA_DEPTH_STREAM = 2,
675     OB_MEDIA_IR_STREAM = 4,
676     OB_MEDIA_GYRO_STREAM = 8,
677     OB_MEDIA_ACCEL_STREAM = 16,
678     OB_MEDIA_CAMERA_PARAM = 32,
679     OB_MEDIA_DEVICE_INFO = 64,
680     OB_MEDIA_STREAM_INFO = 128,
681     OB_MEDIA_IR_LEFT_STREAM = 256,
682     OB_MEDIA_IR_RIGHT_STREAM = 512,
683 } OBMediaType,
684     ob_media_type, OB_MEDIA_TYPE;
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706

```

```

710 typedef enum {
711     OB_MEDIA_BEGIN = 0,
712     OB_MEDIA_PAUSE,
713     OB_MEDIA_RESUME,
714     OB_MEDIA_END,
715 } OBMediaState,
716     ob_media_state, OB_MEDIA_STATE_EM;
717
723 typedef enum {
724     OB_PRECISION_1MM,
725     OB_PRECISION_0MM8,
726     OB_PRECISION_0MM4,
727     OB_PRECISION_0MM1,
728     OB_PRECISION_0MM2,
729     OB_PRECISION_0MM5,
730     OB_PRECISION_0MM05,
731     OB_PRECISION_UNKNOWN,
732     OB_PRECISION_COUNT,
733 } OBDepthPrecisionLevel,
734     ob_depth_precision_level, OB_DEPTH_PRECISION_LEVEL, OBDepthUnit, ob_depth_unit;
735
740 typedef struct {
741     double zpd;          // the distance to calib plane
742     double zpps;          // zpps=z0/fx
743     float baseline;       // baseline length, for monocular camera,it means the distance of laser to the center of
744                     IR-CMOS
745     double fx;           // focus
746     uint8_t bitSize;      // disparity bit size (raw disp bit size, for example: MX6000 is 12, MX6600 is 14)
747     float unit;          // reference units: unit=10 denote 1cm; unit=1 denote 1mm; unit=0.5 denote 0.5mm; and
748                     so on
749     float minDisparity;   // dual disparity coefficient
750     uint8_t packMode;     // data pack mode
751     float dispOffset;     // disparity offset, actual disp=chip disp + disp_offset
752     int32_t invalidDisp;  // invalid disparity, usually is 0, dual IR add a auxiliary value.
753     int32_t dispIntPlace; // disp integer digits, default is 8, Gemini2 XL is 10
754     uint8_t isDualCamera; // 0 monocular camera, 1 dual camera
755 } OBDisparityParam, ob_disparity_param;
756
758 typedef enum {
759     OB_TOF_FILTER_RANGE_CLOSE = 0,
760     OB_TOF_FILTER_RANGE_MIDDLE = 1,
761     OB_TOF_FILTER_RANGE_LONG = 2,
762     OB_TOF_FILTER_RANGE_DEBUG = 100,
763 } OBTofFilterRange,
764     ob_tof_filter_range, TOF_FILTER_RANGE;
765
768 typedef struct {
769     float x;
770     float y;
771     float z;
772 } OBPoint, ob_point, OBPoint3f, ob_point3f;
773
777 typedef struct {
778     float x;
779     float y;
780 } OBPoint2f, ob_point2f;
781
782 typedef struct {
783     float *xTable;
784     float *yTable;
785     int width;
786     int height;
787 } OBXYTables, ob_xy_tables;
788
792 typedef struct {
793     float x;
794     ...
795 }

```

```

194     float y;
195     float z;
196     float r;
197     float g;
198     float b;
199 } OBColorPoint, ob_color_point;
200
204 typedef enum {
205     OB_COMPRESSION LOSSLESS = 0,
206     OB_COMPRESSION LOSSY = 1,
207 } OBCCompressionMode,
208     ob_compression_mode, OB_COMPRESSION_MODE;
209
213 typedef struct {
217     int threshold;
218 } OBCCompressionParams, ob_compression_params, OB_COMPRESSION_PARAMS;
219
223 typedef struct {
224     int32_t upper;
225     int32_t lower;
226 } OTOfExposureThresholdControl, ob_toe_exposure_threshold_control, TOF_EXPOSURE_THRESHOLD_CONTROL;
227
232 typedef enum {
233     OB_SYNC_MODE_CLOSE = 0x00,
234
235     OB_SYNC_MODE_STANDALONE = 0x01,
236
237     OB_SYNC_MODE_PRIMARY = 0x02,
238
239     OB_SYNC_MODE_SECONDARY = 0x03,
240
241     OB_SYNC_MODE_PRIMARY MCU_TRIGGER = 0x04,
242
243     OB_SYNC_MODE_PRIMARY_IR_TRIGGER = 0x05,
244
245     OB_SYNC_MODE_PRIMARY_SOFT_TRIGGER = 0x06,
246
247     OB_SYNC_MODE_SECONDARY_SOFT_TRIGGER = 0x07,
248
249     OB_SYNC_MODE_IR_IMU_SYNC = 0x08,
250
251     OB_SYNC_MODE_UNKNOWN = 0xff,
252
253 } OBSyncMode,
254     ob_sync_mode, OB_SYNC_MODE;
255
259 typedef struct {
260     OBSyncMode syncMode;
261
262     uint16_t irTriggerSignalInDelay;
263
264     uint16_t rgbTriggerSignalInDelay;
265
266     uint16_t deviceTriggerSignalOutDelay;
267
268     uint16_t deviceTriggerSignalOutPolarity;
269
270     uint16_t mcuTriggerFrequency;
271
272     uint16_t deviceId;
273
274 } OBDeviceSyncConfig, ob_device_sync_config, OB_DEVICE_SYNC_CONFIG;
275
279 typedef enum {
280     OB_DEVICE_DEPTH_WORK_MODE = 0

```

```

    OB_DEVICE_DEPTH_WORK_MODE v,
971 OB_CUSTOM_DEPTH_WORK_MODE=1,
972 } OBDepthWorkModeTag,
973 ob_depth_work_mode;
974
975 typedef struct {
976     uint8_t checksum[16];
977
978     char name[32];
979     OBDepthWorkModeTag tag;
980
981 } OBDepthWorkMode, ob_depth_work_mode;
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064

```

```

1068     uint8_t patch;
1069 } OBProtocolVersion, ob_protocol_version;
1070
1074 typedef enum {
1075     OB_CMD_VERSION_V0=(uint16_t)0,
1076     OB_CMD_VERSION_V1=(uint16_t)1,
1077     OB_CMD_VERSION_V2=(uint16_t)2,
1078     OB_CMD_VERSION_V3=(uint16_t)3,
1079
1080     OB_CMD_VERSION_NOVERSION=(uint16_t)0xffff,
1081     OB_CMD_VERSION_INVALID =(uint16_t)0xffff,
1082 } OB_CMD_VERSION,
1083 OBCmdVersion, ob_cmd_version;
1084
1088 typedef struct {
1094     uint16_t dhcp;
1095
1099     uint8_t address[4];
1100
1104     uint8_t mask[4];
1105
1109     uint8_t gateway[4];
1110 } OBNetIpConfig, ob_net_ip_config, DEVICE_IP_ADDR_CONFIG;
1111
1112 #define OBDeviceIpAddrConfig OBNetIpConfig
1113 #define ob_device_ip_addr_config OBNetIpConfig
1114
1118 typedef enum {
1119     OB_COMM_USB=0x00,
1120     OB_COMM_NET=0x01,
1121 } OBCommunicationType,
1122 ob_communication_type, OB_COMMUNICATION_TYPE;
1123
1127 typedef enum {
1128     OB_USB_POWER_NO_PLUGIN=0,
1129     OB_USB_POWER_5V_0A9 =1,
1130     OB_USB_POWER_5V_1A5 =2,
1131     OB_USB_POWER_5V_3A0 =3,
1132 } OBUSBPowerState,
1133 ob_usb_power_state;
1134
1138 typedef enum {
1139     OB_DC_POWER_NO_PLUGIN=0,
1140     OB_DC_POWER_PLUGIN =1,
1141 } OBDCPowerState,
1142 ob_dc_power_state;
1143
1147 typedef enum {
1148     OB_ROTATE_DEGREE_0 =0,
1149     OB_ROTATE_DEGREE_90 =90,
1150     OB_ROTATE_DEGREE_180=180,
1151     OB_ROTATE_DEGREE_270=270,
1152 } ob_rotate_degree_type,
1153 OBRotateDegreeType;
1154
1158 typedef enum {
1159     OB_POWER_LINE_FREQ_MODE_CLOSE=0,
1160     OB_POWER_LINE_FREQ_MODE_50HZ =1,
1161     OB_POWER_LINE_FREQ_MODE_60HZ =2,
1162 } ob_power_line_freq_mode,
1163 OBPowerLineFreqMode;
1164
1168 typedef enum {
1172     OB_FRAME_AGGREGATE_OUTPUT_ALL_TYPE_FRAME_REQUIRE=0,
1173

```

```

1180     OB_FRAME_AGGREGATE_OUTPUT_COLOR_FRAME_REQUIRE,
1181
1187     OB_FRAME_AGGREGATE_OUTPUT_ANY_SITUATION,
1193     OB_FRAME_AGGREGATE_OUTPUT_DISABLE,
1194 } OB_FRAME_AGGREGATE_OUTPUT_MODE,
1195     OBFrameAggregateOutputMode, ob_frame_aggregate_output_mode;
1196 #define OB_FRAME_AGGREGATE_OUTPUT_FULL_FRAME_REQUIRE OB_FRAME_AGGREGATE_O
1197     UTPUT_ALL_TYPE_FRAME_REQUIRE
1198
1199 typedef enum {
1200     OB_LEFT_HAND_COORDINATE_SYSTEM = 0,
1201     OB_RIGHT_HAND_COORDINATE_SYSTEM = 1,
1202 } OB_COORDINATE_SYSTEM_TYPE,
1203     OBCoordinateSystemType, ob_coordinate_system_type;
1204
1205 typedef enum {
1206     OB_USER_MODE = 0,
1207
1208     OB_DEVELOPER_MODE = 1,
1209 } OB_DEVICE DEVELOPMENT MODE,
1210     OBDeviceDevelopmentMode, ob_device_development_mode;
1211
1212 typedef enum {
1213
1214     OB_MULTI_DEVICE_SYNC_MODE_FREE_RUN = 1 << 0,
1215
1216     OB_MULTI_DEVICE_SYNC_MODE_STANDALONE = 1 << 1,
1217
1218     OB_MULTI_DEVICE_SYNC_MODE_PRIMARY = 1 << 2,
1219
1220     OB_MULTI_DEVICE_SYNC_MODE_SECONDARY = 1 << 3,
1221
1222     OB_MULTI_DEVICE_SYNC_MODE_SECONDARY_SYNCED = 1 << 4,
1223
1224     OB_MULTI_DEVICE_SYNC_MODE_SOFTWARE_TRIGGERING = 1 << 5,
1225
1226     OB_MULTI_DEVICE_SYNC_MODE_HARDWARE_TRIGGERING = 1 << 6,
1227
1228     OB_MULTI_DEVICE_SYNC_MODE_IR_IMU_SYNC = 1 << 7,
1229
1230 } ob_multi_device_sync_mode,
1231     OBMultiDeviceSyncMode;
1232
1233 typedef struct {
1234     OBMultiDeviceSyncMode syncMode;
1235
1236     int depthDelayUs;
1237
1238     int colorDelayUs;
1239
1240     int trigger2ImageDelayUs;
1241
1242     bool triggerOutEnable;
1243
1244     int triggerOutDelayUs;
1245
1246     int framesPerTrigger;
1247 } ob_multi_device_sync_config, OBMultiDeviceSyncConfig;
1248
1249 typedef struct {
1250     bool enable;
1251
1252     int timestamp_reset_delay_us;
1253
1254     bool timestamp_reset_signal_output_enable;
1255
1256     ...
1257 }
```

```

1403 } ob_device_timestamp_reset_config, OBDeviceTimestampResetConfig;
1404
1408 typedef struct {
1412     float baseline;
1416     float zpd;
1417 } BASELINE_CALIBRATION_PARAM, ob_baseline_calibration_param, OBBaselineCalibrationParam;
1418
1422 typedef struct {
1423
1431     uint8_t enable;
1432     uint8_t sequence_name;
1433     uint32_t exposure_1;
1434     uint32_t gain_1;
1435     uint32_t exposure_2;
1436     uint32_t gain_2;
1437 } HDR_CONFIG, ob_hdr_config, OBHdrConfig;
1438
1442 typedef struct {
1443     int16_t x0_left;
1444     int16_t y0_top;
1445     int16_t xl_right;
1446     int16_t y1_bottom;
1447 } AE_ROI, ob_region_of_interest, OBRegionOfInterest;
1448
1449 typedef enum {
1450     OB_FILTER_CONFIG_VALUE_TYPE_INVALID=-1,
1451     OB_FILTER_CONFIG_VALUE_TYPE_INT = 0,
1452     OB_FILTER_CONFIG_VALUE_TYPE_FLOAT = 1,
1453     OB_FILTER_CONFIG_VALUE_TYPE_BOOLEAN=2,
1454 } OBFilterConfigValueType,
1455     ob_filter_config_value_type;
1456
1457 typedef struct {
1458     const char      *name;
1459     OBFilterConfigValueType type;
1460     double          min;
1461     double          max;
1462     double          step;
1463     double          def;
1464     const char      *desc;
1465 } OBFilterConfigSchemaItem, ob_filter_config_schema_item;
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535

```

```
1535     OB_FRAME_METADATA_TYPE_EXPOSURE = 1,
1536
1540     OB_FRAME_METADATA_TYPE_WHITE_BALANCE = 7,
1541
1545     OB_FRAME_METADATA_TYPE_BRIGHTNESS = 8,
1546
1550     OB_FRAME_METADATA_TYPE_CONTRAST = 9,
1551
1555     OB_FRAME_METADATA_TYPE_SATURATION = 10,
1556
1560     OB_FRAME_METADATA_TYPE_SHARPNESS = 11,
1561
1565     OB_FRAME_METADATA_TYPE_BACKLIGHT_COMPENSATION = 12,
1566
1570     OB_FRAME_METADATA_TYPE_HUE = 13,
1571
1575     OB_FRAME_METADATA_TYPE_GAMMA = 14,
1576
1581     OB_FRAME_METADATA_TYPE_POWER_LINE_FREQUENCY = 15,
1582
1588     OB_FRAME_METADATA_TYPE_LOW_LIGHT_COMPENSATION = 16,
1589
1593     OB_FRAME_METADATA_TYPE_MANUAL_WHITE_BALANCE = 17,
1594
1599     OB_FRAME_METADATA_TYPE_ACTUAL_FRAME_RATE = 18,
1600
1604     OB_FRAME_METADATA_TYPE_FRAME_RATE = 19,
1605
1609     OB_FRAME_METADATA_TYPE_AE_ROI_LEFT = 20,
1610
1614     OB_FRAME_METADATA_TYPE_AE_ROI_TOP = 21,
1615
1619     OB_FRAME_METADATA_TYPE_AE_ROI_RIGHT = 22,
1620
1624     OB_FRAME_METADATA_TYPE_AE_ROI_BOTTOM = 23,
1625
1629     OB_FRAME_METADATA_TYPE_EXPOSURE_PRIORITY = 24,
1630
1634     OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_NAME = 25,
1635
1639     OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_SIZE = 26,
1640
1644     OB_FRAME_METADATA_TYPE_HDR_SEQUENCE_INDEX = 27,
1645
1651     OB_FRAME_METADATA_TYPE_LASER_POWER = 28,
1652
1656     OB_FRAME_METADATA_TYPE_LASER_POWER_LEVEL = 29,
1657
1662     OB_FRAME_METADATA_TYPE_LASER_STATUS = 30,
1663
1667     OB_FRAME_METADATA_TYPE_GPIO_INPUT_DATA = 31,
1668
1672     OB_FRAME_METADATA_TYPE_DISPARITY_SEARCH_OFFSET = 32,
1673
1677     OB_FRAME_METADATA_TYPE_DISPARITY_SEARCH_RANGE = 33,
1678
1683     OB_FRAME_METADATA_TYPE_COUNT,
1684 } ob_frame_metadata_type,
1685 OBFrameMetadataType;
1686
1691 typedef enum {
1696     OB_UVC_BACKEND_TYPE_AUTO,
1697
1702     OB_UVC_BACKEND_TYPE_LIBUVC,
1703
1708     OB_UVC_BACKEND_TYPE_V4L2,
```

```

1709
1710
1711     OB_UVC_BACKEND_TYPE_MSMF,
1712 } ob_uvc_backend_type,
1713     OBUvcBackendType;
1714
1715
1716
1717     typedef enum {
1718         OB_PLAYBACK_UNKNOWN,
1719         OB_PLAYBACK_PLAYING,
1720         OB_PLAYBACK_PAUSED,
1721         OB_PLAYBACK_STOPPED,
1722         OB_PLAYBACK_COUNT,
1723     } ob_playback_status,
1724     OBPlaybackStatus;
1725
1726
1727
1728 // For compatibility
1729 #define OB_FRAME_METADATA_TYPE_LASER_POWER_MODE OB_FRAME_METADATA_TYPE
1730     LASER_POWER_LEVEL
1731 #define OB_FRAME_METADATA_TYPE_EMITTER_MODE OB_FRAME_METADATA_TYPE_LASE
1732     R_STATUS
1733
1734
1735
1736     typedef void (*ob_file_send_callback)(ob_file_tran_state state, const char *message, uint8_t percent, voi
1737         d *user_data);
1738
1739
1740     typedef void (*ob_device_fw_update_callback)(ob_fw_update_state state, const char *message, uint8_t
1741         percent, void *user_data);
1742
1743
1744     typedef void (*ob_device_state_callback)(ob_device_state state, const char *message, void *user_data)
1745         ;
1746
1747
1748     typedef void (*ob_set_data_callback)(ob_data_tran_state state, uint8_t percent, void *user_data);
1749
1750
1751     typedef void (*ob_get_data_callback)(ob_data_tran_state state, ob_data_chunk *dataChunk, void *user
1752         _data);
1753
1754
1755     typedef void (*ob_media_state_callback)(ob_media_state state, void *user_data);
1756
1757
1758     typedef void (*ob_device_changed_callback)(ob_device_list *removed, ob_device_list *added, void *us
1759         er_data);
1760
1761
1762 // typedef void (*ob_net_device_added_callback)(const char *added, void *user_data);
1763 // typedef void (*ob_net_device_removed_callback)(const char *removed, void *user_data);
1764
1765
1766     typedef void (*ob_frame_callback)(ob_frame *frame, void *user_data);
1767 #define ob_filter_callback ob_frame_callback
1768 #define ob_playback_callback ob_frame_callback
1769
1770
1771     typedef void (*ob_frameset_callback)(ob_frame *frameset, void *user_data);
1772
1773
1774     typedef void(ob_frame_destroy_callback)(uint8_t *buffer, void *user_data);
1775
1776
1777     typedef void(ob_log_callback)(ob_log_severity severity, const char *message, void *user_data);
1778
1779
1780     typedef void (*ob_playback_status_changed_callback)(ob_playback_status status, void *user_data);
1781 #define ob_is_video_sensor_type(sensor_type)
1782     (sensor_type == OB_SENSOR_COLOR || sensor_type == OB_SENSOR_DEPTH || sensor_type == OB_S
1783         ENSOR_IR || sensor_type == OB_SENSOR_IR_LEFT \
1784     || sensor_type == OB_SENSOR_IR_RIGHT || sensor_type == OB_SENSOR_CONFIDENCE)
1785
1786
1787 #define ob_is_video_stream_type(stream_type)
1788     (stream_type == OB_STREAM_COLOR || stream_type == OB_STREAM_DEPTH || stream_type == OB_
1789         STREAM_IR || stream_type == OB_STREAM_IR_LEFT \
1790     || stream_type == OB_STREAM_IR_RIGHT || stream_type == OB_STREAM_VIDEO || stream_type ==
1791         OB_STREAM_CONFIDENCE)
1792
1793
1794 #define is_ir_sensor(sensor_type) (sensor_type == OB_SENSOR_IR || sensor_type == OB_SENSOR_IR_

```

```
    LEFT || sensor_type == OB_SENSOR_IR_RIGHT)
1863 #define isIRSensor is_ir_sensor
1864
1871 #define is_ir_stream(stream_type) (stream_type == OB_STREAM_IR || stream_type == OB_STREAM_IR
1872     _LEFT || stream_type == OB_STREAM_IR_RIGHT)
1873 #define isIRStream is_ir_stream
1874
1880 #define is_ir_frame(frame_type) (frame_type == OB_FRAME_IR || frame_type == OB_FRAME_IR_LEFT ||
1881     frame_type == OB_FRAME_IR_RIGHT)
1882 #define isIRFrame is_ir_frame
1883
1886 #define OB_DEFAULT_DECRYPT_KEY(nullptr)
1887
1888 #ifdef __cplusplus
1889 }
1890 #endif
1891
1892 #pragma pack(pop)
```

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

# Pipeline.h File Reference

The SDK's advanced API can quickly implement functions such as switching streaming, frame synchronization, software filtering, etc., suitable for applications, and the algorithm focuses on rgbd data stream scenarios. If you are on real-time or need to handle synchronization separately, align the scene. Please use the interface of Device's Lower API. [More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_config_set_depth_scale_require ob_config_set_depth_scale_after_align_require
```

## Functions

```
OB_EXPORT ob_pipeline * ob_create_pipeline (ob_error **error)
```

Create a pipeline object.

```
OB_EXPORT ob_pipeline * ob_create_pipeline_with_device (const ob_device *dev,  
ob_error **error)
```

Using device objects to create pipeline objects.

```
OB_EXPORT void ob_delete_pipeline (ob_pipeline *pipeline, ob_error **error)
```

Delete pipeline objects.

```
OB_EXPORT void ob_pipeline_start (ob_pipeline *pipeline, ob_error **error)
```

Start the pipeline with default parameters.

```
OB_EXPORT void ob_pipeline_start_with_config (ob_pipeline *pipeline, const  
ob_config *config, ob_error **error)
```

Start the pipeline with configuration parameters.

```
OB_EXPORT void ob_pipeline_start_with_callback (ob_pipeline *pipeline, const  
ob_config *config, ob_frameset_callback callback, void  
*user_data, ob_error **error)
```

Start the pipeline and set the frame collection data callback.

```
OB_EXPORT void ob_pipeline_stop (ob_pipeline *pipeline, ob_error **error)
```

Stop pipeline.

```
OB_EXPORT ob_config * ob_pipeline_get_config (const ob_pipeline *pipeline,  
ob_error **error)
```

Get the configuration object associated with the pipeline.

```
OB_EXPORT void ob_pipeline_switch_config (ob_pipeline *pipeline, ob_config  
*config, ob_error **error)
```

Switch the corresponding configuration.

**OB\_EXPORT** **ob\_frame** \* **ob\_pipeline\_wait\_for\_frameset** (**ob\_pipeline** \*pipeline,  
                  **uint32\_t** timeout\_ms, **ob\_error** \*\*error)

Wait for a set of frames to be returned synchronously.

**OB\_EXPORT** **ob\_device** \* **ob\_pipeline\_get\_device** (const **ob\_pipeline** \*pipeline,  
                  **ob\_error** \*\*error)

Get the device object associated with the pipeline.

**OB\_EXPORT** **ob\_stream\_profile\_list** \* **ob\_pipeline\_get\_stream\_profile\_list** (const **ob\_pipeline**  
                  \*bipeline, **ob\_sensor\_type** sensorType, **ob\_error** \*\*error)

Get the stream profile list associated with the pipeline.

**OB\_EXPORT** void **ob\_pipeline\_enable\_frame\_sync** (**ob\_pipeline** \*pipeline,  
                  **ob\_error** \*\*error)

Enable frame synchronization.

**OB\_EXPORT** void **ob\_pipeline\_disable\_frame\_sync** (**ob\_pipeline** \*pipeline,  
                  **ob\_error** \*\*error)

Disable frame synchronization.

**OB\_EXPORT** **ob\_stream\_profile\_list** \* **ob\_get\_d2c\_depth\_profile\_list** (const **ob\_pipeline** \*pipeline,  
                  const **ob\_stream\_profile** \*color\_profile, **ob\_align\_mode**  
                  align\_mode, **ob\_error** \*\*error)

Return a list of D2C-enabled depth sensor resolutions  
corresponding to the input color sensor resolution.

**OB\_EXPORT** **ob\_config** \* **ob\_create\_config** (**ob\_error** \*\*error)

Create the pipeline configuration.

**OB\_EXPORT** void **ob\_delete\_config** (**ob\_config** \*config, **ob\_error** \*\*error)

Delete the pipeline configuration.

**OB\_EXPORT** void **ob\_config\_enable\_stream** (**ob\_config** \*config,  
                  **ob\_stream\_type** stream\_type, **ob\_error** \*\*error)

Enable a stream with default profile.

**OB\_EXPORT** void **ob\_config\_enable\_all\_stream** (**ob\_config** \*config, **ob\_error**  
                  \*\*error)

Enable all streams in the pipeline configuration.

**OB\_EXPORT** void **ob\_config\_enable\_stream\_with\_stream\_profile** (**ob\_config**  
                  \*config, const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Enable a stream according to the stream profile.

**OB\_EXPORT** void **ob\_config\_enable\_video\_stream** (**ob\_config** \*config,  
                  **ob\_stream\_type** stream\_type, **uint32\_t** width, **uint32\_t** height,  
                  **uint32\_t** fps, **ob\_format** format, **ob\_error** \*\*error)

Enable video stream with specified parameters.

**OB\_EXPORT** void **ob\_config\_enable\_accel\_stream** (**ob\_config** \*config,  
**ob\_accel\_full\_scale\_range** full\_scale\_range,  
**ob\_accel\_sample\_rate** sample\_rate, **ob\_error** \*\*error)

Enable accelerometer stream with specified parameters.

**OB\_EXPORT** void **ob\_config\_enable\_gyro\_stream** (**ob\_config** \*config,  
**ob\_gyro\_full\_scale\_range** full\_scale\_range,  
**ob\_gyro\_sample\_rate** sample\_rate, **ob\_error** \*\*error)

Enable gyroscope stream with specified parameters.

**OB\_EXPORT** **ob\_stream\_profile\_list** \* **ob\_config\_get\_enabled\_stream\_profile\_list** (const **ob\_config** \*config, **ob\_error** \*\*error)

Get the enabled stream profile list in the pipeline configuration.

**OB\_EXPORT** void **ob\_config\_disable\_stream** (**ob\_config** \*config, **ob\_stream\_type** type, **ob\_error** \*\*error)

Disable a specific stream in the pipeline configuration.

**OB\_EXPORT** void **ob\_config\_disable\_all\_stream** (**ob\_config** \*config, **ob\_error** \*\*error)

Disable all streams in the pipeline configuration.

**OB\_EXPORT** void **ob\_config\_set\_align\_mode** (**ob\_config** \*config, **ob\_align\_mode** mode, **ob\_error** \*\*error)

Set the alignment mode for the pipeline configuration.

**OB\_EXPORT** void **ob\_config\_set\_depth\_scale\_after\_align\_require** (**ob\_config** \*config, bool enable, **ob\_error** \*\*error)

Set whether depth scaling is required after enable depth to color alignment.

**OB\_EXPORT** void **ob\_config\_set\_frame\_aggregate\_output\_mode** (**ob\_config** \*config, **ob\_frame\_aggregate\_output\_mode** mode, **ob\_error** \*\*error)

Set the frame aggregation output mode for the pipeline configuration.

**OB\_EXPORT** **ob\_camera\_param** **ob\_pipeline\_get\_camera\_param** (**ob\_pipeline** \*pipeline, **ob\_error** \*\*error)

Get current camera parameters.

**OB\_EXPORT** **ob\_camera\_param** **ob\_pipeline\_get\_camera\_param\_with\_profile** (**ob\_pipeline** \*pipeline, uint32\_t colorWidth, uint32\_t colorHeight, uint32\_t depthWidth, uint32\_t depthHeight, **ob\_error** \*\*error)

Get the current camera parameters.

**OB\_EXPORT** **ob\_calibration\_param** **ob\_pipeline\_get\_calibration\_param** (**ob\_pipeline** \*pipeline, **ob\_config** \*config, **ob\_error** \*\*error)

Get device calibration parameters with the specified configuration.

## Detailed Description

The SDK's advanced API can quickly implement functions such as switching streaming, frame synchronization, software filtering, etc., suitable for applications, and the algorithm focuses on rgbd data stream scenarios. If you are on real-time or need to handle synchronization separately, align the scene. Please use the interface of Device's Lower API.

Definition in file [Pipeline.h](#).

## Macro Definition Documentation

- ◆ [ob\\_config\\_set\\_depth\\_scale\\_require](#)

```
#define ob_config_set_depth_scale_require ob\_config\_set\_depth\_scale\_after\_align\_require
```

Definition at line [330](#) of file [Pipeline.h](#).

## Function Documentation

- ◆ [ob\\_create\\_pipeline\(\)](#)

```
OB_EXPORT ob\_pipeline * ob_create_pipeline ( ob\_error ** error )
```

Create a pipeline object.

### Parameters

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

[ob\\_pipeline](#)\* return the pipeline object

Referenced by [ob::Pipeline::Pipeline\(\)](#).

- ◆ [ob\\_create\\_pipeline\\_with\\_device\(\)](#)

```
OB_EXPORT ob_pipeline* ob_create_pipeline_with_device ( const ob_device* dev,  
                                                       ob_error** error )
```

Using device objects to create pipeline objects.

### Parameters

- [in] **dev** Device object used to create pipeline
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

ob\_pipeline\* return the pipeline object

Referenced by [ob::Pipeline::Pipeline\(\)](#).

### ◆ [ob\\_delete\\_pipeline\(\)](#)

```
OB_EXPORT void ob_delete_pipeline ( ob_pipeline* pipeline,  
                                    ob_error** error )
```

Delete pipeline objects.

### Parameters

- [in] **pipeline** The pipeline object to be deleted
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Pipeline::~Pipeline\(\)](#).

### ◆ [ob\\_pipeline\\_start\(\)](#)

```
OB_EXPORT void ob_pipeline_start ( ob_pipeline* pipeline,  
                                    ob_error** error )
```

Start the pipeline with default parameters.

### Parameters

- [in] **pipeline** pipeline object
- [out] **error** Pointer to an error object that will be set if an error occurs.

### ◆ [ob\\_pipeline\\_start\\_with\\_config\(\)](#)

```
OB_EXPORT void ob_pipeline_start_with_config ( ob_pipeline * pipeline,  
                                              const ob_config * config,  
                                              ob_error ** error )
```

Start the pipeline with configuration parameters.

### Parameters

- [in] **pipeline** pipeline object
- [in] **config** Parameters to be configured
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Pipeline::start\(\)](#).

### ◆ [ob\\_pipeline\\_start\\_with\\_callback\(\)](#)

```
OB_EXPORT void ob_pipeline_start_with_callback ( ob_pipeline * pipeline,  
                                                const ob_config * config,  
                                                ob_frameset_callback callback,  
                                                void * user_data,  
                                                ob_error ** error )
```

Start the pipeline and set the frame collection data callback.

### Attention

After start the pipeline with this interface, the frames will be output to the callback function and cannot be obtained frames by call `@ob_pipeline_wait_for_frameset`

### Parameters

- [in] **pipeline** pipeline object
- [in] **config** Parameters to be configured
- [in] **callback** Trigger a callback when all frame data in the frameset arrives
- [in] **user\_data** Pass in any user data and get it from the callback
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Pipeline::start\(\)](#).

### ◆ [ob\\_pipeline\\_stop\(\)](#)

```
OB_EXPORT void ob_pipeline_stop ( ob_pipeline * pipeline,  
                                ob_error ** error )
```

Stop pipeline.

#### Parameters

[in] **pipeline** pipeline object

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::Pipeline::stop()**.

#### ◆ **ob\_pipeline\_get\_config()**

```
OB_EXPORT ob_config * ob_pipeline_get_config ( const ob_pipeline * pipeline,  
                                              ob_error ** error )
```

Get the configuration object associated with the pipeline.

Returns default configuration if the user has not configured

#### Parameters

[in] **pipeline** The pipeline object

[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_config**\* The configuration object

Referenced by **ob::Pipeline::getConfig()**.

#### ◆ **ob\_pipeline\_switch\_config()**

```
OB_EXPORT void ob_pipeline_switch_config ( ob_pipeline * pipeline,  
                                         ob_config * config,  
                                         ob_error ** error )
```

Switch the corresponding configuration.

#### Parameters

- [in] **pipeline** The pipeline object
- [in] **config** The pipeline configuration
- [out] **error** Log error messages

#### ◆ **ob\_pipeline\_wait\_for\_frameset()**

```
OB_EXPORT ob_frame * ob_pipeline_wait_for_frameset ( ob_pipeline * pipeline,  
                                                 uint32_t      timeout_ms,  
                                                 ob_error ** error )
```

Wait for a set of frames to be returned synchronously.

#### Parameters

- [in] **pipeline** The pipeline object
- [in] **timeout\_ms** The timeout for waiting (in milliseconds)
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_frame\*** The frameset that was waited for. A frameset is a special frame that can be used to obtain independent frames from the set.

Referenced by **ob::Pipeline::waitForFrameset()**.

#### ◆ **ob\_pipeline\_get\_device()**

```
OB_EXPORT ob_device * ob_pipeline_get_device ( const ob_pipeline * pipeline,  
                                              ob_error **          error )
```

Get the device object associated with the pipeline.

#### Parameters

- [in] **pipeline** The pipeline object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

ob\_device\* The device object

Referenced by [ob::Pipeline::getDevice\(\)](#).

#### ◆ [ob\\_pipeline\\_get\\_stream\\_profile\\_list\(\)](#)

```
OB_EXPORT ob_stream_profile_list * ob_pipeline_get_stream_profile_list ( const ob_pipeline * pipeline,  
                           ob_sensor_type      sensorType,  
                           ob_error **          error )
```

Get the stream profile list associated with the pipeline.

#### Parameters

- [in] **pipeline** The pipeline object
- [in] **sensorType** The sensor type. The supported sensor types can be obtained through the [ob\\_device\\_get\\_sensor\\_list\(\)](#) interface.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

ob\_stream\_profile\_list\* The stream profile list

Referenced by [ob::Pipeline::getStreamProfileList\(\)](#).

#### ◆ [ob\\_pipeline\\_enable\\_frame\\_sync\(\)](#)

```
OB_EXPORT void ob_pipeline_enable_frame_sync ( ob_pipeline * pipeline,  
                                              ob_error ** error )
```

Enable frame synchronization.

Synchronize the frames of different streams by using the timestamp information of the frames.

Dynamically (when pipeline is started) enable/disable frame synchronization is allowed.

### Parameters

[in] **pipeline** The pipeline object

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Pipeline::enableFrameSync\(\)](#).

- ◆ [ob\\_pipeline\\_disable\\_frame\\_sync\(\)](#)

```
OB_EXPORT void ob_pipeline_disable_frame_sync ( ob_pipeline * pipeline,  
                                              ob_error ** error )
```

Disable frame synchronization.

### Parameters

[in] **pipeline** The pipeline object

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Pipeline::disableFrameSync\(\)](#).

- ◆ [ob\\_get\\_d2c\\_depth\\_profile\\_list\(\)](#)

```
OB_EXPORT ob_stream_profile_list* ob_get_d2c_depth_profile_list ( const ob_pipeline * pipeline,  
                                         const ob_stream_profile * color_profile,  
                                         ob_align_mode align_mode,  
                                         ob_error ** error )
```

Return a list of D2C-enabled depth sensor resolutions corresponding to the input color sensor resolution.

### Parameters

[in] **pipeline** The pipeline object  
[in] **color\_profile** The input profile of the color sensor  
[in] **align\_mode** The input align mode  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_stream\_profile\_list**\* The list of D2C-enabled depth sensor resolutions

Referenced by **ob::Pipeline::getD2CDepthProfileList()**.

### ◆ **ob\_create\_config()**

```
OB_EXPORT ob_config * ob_create_config ( ob_error ** error )
```

Create the pipeline configuration.

### Parameters

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_config**\* The configuration object

Referenced by **ob::Config::Config()**.

### ◆ **ob\_delete\_config()**

```
OB_EXPORT void ob_delete_config ( ob_config * config,  
                                ob_error ** error )
```

Delete the pipeline configuration.

#### Parameters

- [in] **config** The configuration to be deleted
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::~Config\(\)](#).

#### ◆ [ob\\_config\\_enable\\_stream\(\)](#)

```
OB_EXPORT void ob_config_enable_stream ( ob_config * config,  
                                         ob_stream_type stream_type,  
                                         ob_error ** error )
```

Enable a stream with default profile.

#### Parameters

- [in] **config** The pipeline configuration object
- [in] **stream\_type** The type of the stream to be enabled
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::enableStream\(\)](#).

#### ◆ [ob\\_config\\_enable\\_all\\_stream\(\)](#)

```
OB_EXPORT void ob_config_enable_all_stream ( ob_config * config,  
                                              ob_error ** error )
```

Enable all streams in the pipeline configuration.

#### Parameters

- [in] **config** The pipeline configuration
- [out] **error** Log error messages

Referenced by [ob::Config::enableAllStream\(\)](#).

#### ◆ [ob\\_config\\_enable\\_stream\\_with\\_stream\\_profile\(\)](#)

```
OB_EXPORT void ob_config_enable_stream_with_stream_profile( ob_config * config,  
                                              const ob_stream_profile * profile,  
                                              ob_error ** error )
```

Enable a stream according to the stream profile.

#### Parameters

- [in] **config** The pipeline configuration object
- [in] **profile** The stream profile to be enabled
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::enableStream\(\)](#).

#### ◆ [ob\\_config\\_enable\\_video\\_stream\(\)](#)

```
OB_EXPORT void ob_config_enable_video_stream( ob_config * config,  
                                              ob_stream_type stream_type,  
                                              uint32_t width,  
                                              uint32_t height,  
                                              uint32_t fps,  
                                              ob_format format,  
                                              ob_error ** error )
```

Enable video stream with specified parameters.

#### Attention

The stream\_type should be a video stream type, such as OB\_STREAM\_IR, OB\_STREAM\_COLOR, OB\_STREAM\_DEPTH, etc.

#### Parameters

- [in] **config** The pipeline configuration object
- [in] **stream\_type** The type of the stream to be enabled
- [in] **width** The width of the video stream
- [in] **height** The height of the video stream
- [in] **fps** The frame rate of the video stream
- [in] **format** The format of the video stream
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::enableVideoStream\(\)](#).

◆ [ob\\_config\\_enable\\_accel\\_stream\(\)](#)

```
OB_EXPORT void ob_config_enable_accel_stream( ob_config * config,  
                                              ob_accel_full_scale_range full_scale_range,  
                                              ob_accel_sample_rate sample_rate,  
                                              ob_error ** error )
```

Enable accelerometer stream with specified parameters.

**Parameters**

[in] <b>config</b>	The pipeline configuration object
[in] <b>full_scale_range</b>	The full scale range of the accelerometer
[in] <b>sample_rate</b>	The sample rate of the accelerometer
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::enableAccelStream\(\)](#).

◆ [ob\\_config\\_enable\\_gyro\\_stream\(\)](#)

```
OB_EXPORT void ob_config_enable_gyro_stream( ob_config * config,  
                                              ob_gyro_full_scale_range full_scale_range,  
                                              ob_gyro_sample_rate sample_rate,  
                                              ob_error ** error )
```

Enable gyroscope stream with specified parameters.

**Parameters**

[in] <b>config</b>	The pipeline configuration object
[in] <b>full_scale_range</b>	The full scale range of the gyroscope
[in] <b>sample_rate</b>	The sample rate of the gyroscope
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::enableGyroStream\(\)](#).

◆ [ob\\_config\\_get\\_enabled\\_stream\\_profile\\_list\(\)](#)

Get the enabled stream profile list in the pipeline configuration.

## Parameters

**config** The pipeline configuration object

**error** Pointer to an error object that will be set if an error occurs.

## Returns

`ob_stream_profile_list*` The enabled stream profile list, should be released by `ob_delete_stream_profile_list` after use

Referenced by [ob::Config::getEnabledStreamProfileList\(\)](#).

#### ◆ ob config disable stream()

Disable a specific stream in the pipeline configuration.

## Parameters

[in] **config** The pipeline configuration object

[in] **type** The type of stream to be disabled

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::disableStream\(\)](#).

## ◆ ob\_config\_disable\_all\_stream()

```
OB_EXPORT void ob_config_disable_all_stream( ob_config * config,  
                                              ob_error ** error )
```

Disable all streams in the pipeline configuration.

#### Parameters

- [in] **config** The pipeline configuration object
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::disableAllStream\(\)](#).

#### ◆ [ob\\_config\\_set\\_align\\_mode\(\)](#)

```
OB_EXPORT void ob_config_set_align_mode( ob_config * config,  
                                         ob_align_mode mode,  
                                         ob_error ** error )
```

Set the alignment mode for the pipeline configuration.

#### Parameters

- [in] **config** The pipeline configuration object
- [in] **mode** The alignment mode to be set
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::setAlignMode\(\)](#).

#### ◆ [ob\\_config\\_set\\_depth\\_scale\\_after\\_align\\_require\(\)](#)

```
OB_EXPORT void ob_config_set_depth_scale_after_align_require( ob_config * config,  
                                bool enable,  
                                ob_error ** error )
```

Set whether depth scaling is required after enable depth to color alignment.

After enabling depth to color alignment, the depth image may need to be scaled to match the color image size.

## Parameters

- [in] **config** The pipeline configuration object
- [in] **enable** Whether scaling is required
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::setDepthScaleRequire\(\)](#).

## ◆ [ob\\_config\\_set\\_frame\\_aggregate\\_output\\_mode\(\)](#)

```
OB_EXPORT void ob_config_set_frame_aggregate_output_mode( ob_config * config,  
                                         ob_frame_aggregate_output_mode mode,  
                                         ob_error ** error )
```

Set the frame aggregation output mode for the pipeline configuration.

The processing strategy when the FrameSet generated by the frame aggregation function does not contain the frames of all opened streams (which can be caused by different frame rates of each stream, or by the loss of frames of one stream): drop directly or output to the user.

## Parameters

- [in] **config** The pipeline configuration object
- [in] **mode** The frame aggregation output mode to be set (default mode is [OB\\_FRAME\\_AGGREGATE\\_OUTPUT\\_ANY\\_SITUATION](#))
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::Config::setFrameAggregateOutputMode\(\)](#).

## ◆ [ob\\_pipeline\\_get\\_camera\\_param\(\)](#)

```
OB_EXPORT ob_camera_param ob_pipeline_get_camera_param( ob_pipeline * pipeline,  
                                              ob_error ** error )
```

Get current camera parameters.

### Attention

If D2C is enabled, it will return the camera parameters after D2C, if not, it will return to the default parameters

### Parameters

[in] **pipeline** pipeline object  
[out] **error** Log error messages

### Returns

**ob\_camera\_param** The camera internal parameters

Referenced by **ob::Pipeline::getCameraParam()**.

- ◆ [ob\\_pipeline\\_get\\_camera\\_param\\_with\\_profile\(\)](#)

```
OB_EXPORT ob_camera_param ob_pipeline_get_camera_param_with_profile( ob_pipeline * pipeline,
                                                               uint32_t      colorWidth,
                                                               uint32_t      colorHeight,
                                                               uint32_t      depthWidth,
                                                               uint32_t      depthHeight,
                                                               ob_error ** error )
```

Get the current camera parameters.

### Parameters

- [in] **pipeline** pipeline object
- [in] **colorWidth** color width
- [in] **colorHeight** color height
- [in] **depthWidth** depth width
- [in] **depthHeight** depth height
- [out] **error** Log error messages

### Returns

**ob\_camera\_param** returns camera internal parameters

Referenced by **ob::Pipeline::getCameraParamWithProfile()**.

## ◆ **ob\_pipeline\_get\_calibration\_param()**

```
OB_EXPORT ob_calibration_param ob_pipeline_get_calibration_param( ob_pipeline * pipeline,
                                                               ob_config * config,
                                                               ob_error ** error )
```

Get device calibration parameters with the specified configuration.

### Parameters

- [in] **pipeline** pipeline object
- [in] **config** The pipeline configuration
- [out] **error** Log error messages

### Returns

**ob\_calibration\_param** The calibration parameters

Referenced by **ob::Pipeline::getCalibrationParam()**.



# Pipeline.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
10 #pragma once
11
12 #ifndef __cplusplus
13 extern "C" {
14 #endif
15
16 #include "ObTypes.h"
17
24 OB_EXPORT ob_pipeline *ob_create_pipeline(ob_error **error);
25
33 OB_EXPORT ob_pipeline *ob_create_pipeline_with_device(const ob_device *dev, ob_error **error);
34
41 OB_EXPORT void ob_delete_pipeline(ob_pipeline *pipeline, ob_error **error);
42
49 OB_EXPORT void ob_pipeline_start(ob_pipeline *pipeline, ob_error **error);
50
58 OB_EXPORT void ob_pipeline_start_with_config(ob_pipeline *pipeline, const ob_config *config, ob_error **error);
59
72 OB_EXPORT void ob_pipeline_start_with_callback(ob_pipeline *pipeline, const ob_config *config, ob_frameset_callback callback, void *user_data,
73 ob_error **error);
74
81 OB_EXPORT void ob_pipeline_stop(ob_pipeline *pipeline, ob_error **error);
82
91 OB_EXPORT ob_config *ob_pipeline_get_config(const ob_pipeline *pipeline, ob_error **error);
92
100 OB_EXPORT void ob_pipeline_switch_config(ob_pipeline *pipeline, ob_config *config, ob_error **error);
101
110 OB_EXPORT ob_frame *ob_pipeline_wait_for_frameset(ob_pipeline *pipeline, uint32_t timeout_ms, ob_error **error);
111
119 OB_EXPORT ob_device *ob_pipeline_get_device(const ob_pipeline *pipeline, ob_error **error);
120
129 OB_EXPORT ob_stream_profile_list *ob_pipeline_get_stream_profile_list(const ob_pipeline *pipeline, ob_sensor_type sensorType, ob_error **error);
130
139 OB_EXPORT void ob_pipeline_enable_frame_sync(ob_pipeline *pipeline, ob_error **error);
140
147 OB_EXPORT void ob_pipeline_disable_frame_sync(ob_pipeline *pipeline, ob_error **error);
148
158 OB_EXPORT ob_stream_profile_list *ob_get_d2c_depth_profile_list(const ob_pipeline *pipeline, const ob_stream_profile *color_profile, ob_align_mode align_mode,
159 ob_error **error);
160
167 OB_EXPORT ob_config *ob_create_config(ob_error **error);
168
175 OB_EXPORT void ob_delete_config(ob_config *config, ob_error **error);
176
184 OB_EXPORT void ob_config_enable_stream(ob_config *config, ob_stream_type stream_type, ob_error **error);
185
192 OB_EXPORT void ob_config_enable_all_stream(ob_config *config, ob_error **error);
193
```

```

201 OB_EXPORT void ob_config_enable_stream_with_stream_profile(ob_config *config, const ob_stream_p
202   rofile *profile, ob_error **error);
203
216 OB_EXPORT void ob_config_enable_video_stream(ob_config *config, ob_stream_type stream_type, uint32_t width, uint32_t height, uint32_t fps, ob_format format,
217   ob_error **error);
218
227 OB_EXPORT void ob_config_enable_accel_stream(ob_config *config, ob_accel_full_scale_range full_scale_range, ob_accel_sample_rate sample_rate, ob_error **error);
228
237 OB_EXPORT void ob_config_enable_gyro_stream(ob_config *config, ob_gyro_full_scale_range full_scale_range, ob_gyro_sample_rate sample_rate, ob_error **error);
238
246 OB_EXPORT ob_stream_profile_list *ob_config_get_enabled_stream_profile_list(const ob_config *config, ob_error **error);
247
255 OB_EXPORT void ob_config_disable_stream(ob_config *config, ob_stream_type type, ob_error **error)
256   ;
263 OB_EXPORT void ob_config_disable_all_stream(ob_config *config, ob_error **error);
264
272 OB_EXPORT void ob_config_set_align_mode(ob_config *config, ob_align_mode mode, ob_error **error);
273
282 OB_EXPORT void ob_config_set_depth_scale_after_align_require(ob_config *config, bool enable, ob_error **error);
283
293 OB_EXPORT void ob_config_set_frame_aggregate_output_mode(ob_config *config, ob_frame_aggregate_output_mode mode, ob_error **error);
294
303 OB_EXPORT ob_camera_param ob_pipeline_get_camera_param(ob_pipeline *pipeline, ob_error **error);
304
316 OB_EXPORT ob_camera_param ob_pipeline_get_camera_param_with_profile(ob_pipeline *pipeline, uint32_t colorWidth, uint32_t colorHeight, uint32_t depthWidth,
317   uint32_t depthHeight, ob_error **error);
318
327 OB_EXPORT ob_calibration_param ob_pipeline_get_calibration_param(ob_pipeline *pipeline, ob_config *config, ob_error **error);
328
329 // The following interfaces are deprecated and are retained here for compatibility purposes.
330 #define ob_config_set_depth_scale_require ob_config_set_depth_scale_after_align_require
331
332 #ifdef __cplusplus
333 }
334 #endif
335

```

# Pipeline.hpp File Reference

The SDK's advanced API type can quickly implement switching streaming and frame synchronization operations. [More...](#)

```
#include "Frame.hpp"
#include "Device.hpp"
#include "StreamProfile.hpp"
#include "libobsensor/h/Pipeline.h"
#include "libobsensor.hpp/Types.hpp"
#include "libobsensor.hpp/TypeHelper.hpp"
#include <memory>
#include <functional>
```

[Go to the source code of this file.](#)

## Classes

class **ob::Config**

[Config](#) class for configuring pipeline parameters. [More...](#)

class **ob::Pipeline**

## Namespaces

namespace **ob**

## Detailed Description

The SDK's advanced API type can quickly implement switching streaming and frame synchronization operations.

Definition in file [Pipeline.hpp](#).

# Pipeline.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6
7
8
9
10
11 #include "Frame.hpp"
12 #include "Device.hpp"
13 #include "StreamProfile.hpp"
14
15 #include "libobssensor/h/Pipeline.h"
16 #include "libobssensor/hpp/Types.hpp"
17 #include "libobssensor/hpp/TypeHelper.hpp"
18
19 #include <memory>
20 #include <functional>
21 namespace ob {
22
23     class Config {
24
25     private:
26         ob_config_t *impl_;
27
28     public:
29         Config() {
30             ob_error *error = nullptr;
31             impl_ = ob_create_config(&error);
32             Error::handle(&error);
33         }
34
35         explicit Config(ob_config_t *impl) : impl_(impl) {}
36
37         ~Config() noexcept {
38             ob_error *error = nullptr;
39             ob_delete_config(impl_, &error);
40             Error::handle(&error, false);
41         }
42
43         ob_config_t *getImpl() const {
44             return impl_;
45         }
46
47         void enableStream(OBStreamType streamType) const {
48             ob_error *error = nullptr;
49             ob_config_enable_stream(impl_, streamType, &error);
50             Error::handle(&error);
51         }
52
53         void enableStream(OBSensorType sensorType) const {
54             auto streamType = ob::TypeHelper::convertSensorTypeToStreamType(sensorType);
55             enableStream(streamType);
56         }
57
58         void enableStream(std::shared_ptr<const StreamProfile> streamProfile) const {
59             ob_error *error = nullptr;
60             auto c_stream_profile = streamProfile->getImpl();
61             ob_config_enable_stream_with_stream_profile(impl_, c_stream_profile, &error);
62             Error::handle(&error);
63         }
64
65     };
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
```

```

104 void enableVideoStream(OBStreamType streamType, uint32_t width = OB_WIDTH_ANY, uint32_t heig
105     ht = OB_HEIGHT_ANY, uint32_t fps = OB_FPS_ANY,
106         OBFormat format = OB_FORMAT_ANY) const {
107     ob_error *error = nullptr;
108     ob_config_enable_video_stream(impl_, streamType, width, height, fps, format, &error);
109     Error::handle(&error);
110 }
111
112 void enableVideoStream(OBSensorType sensorType, uint32_t width = OB_WIDTH_ANY, uint32_t heig
113     ht = OB_HEIGHT_ANY, uint32_t fps = OB_FPS_ANY,
114         OBFormat format = OB_FORMAT_ANY) const {
115     auto streamType = ob::TypeHelper::convertSensorTypeToStreamType(sensorType);
116     enableVideoStream(streamType, width, height, fps, format);
117 }
118
119 void enableAccelStream(OBAccelFullScaleRange fullScaleRange = OB_ACCEL_FULL_SCALE_RANGE_
120     _ANY,
121         OBAccelSampleRate sampleRate = OB_ACCEL_SAMPLE_RATE_ANY) const {
122     ob_error *error = nullptr;
123     ob_config_enable_accel_stream(impl_, fullScaleRange, sampleRate, &error);
124     Error::handle(&error);
125 }
126
127 void enableGyroStream(OBGyroFullScaleRange fullScaleRange = OB_GYRO_FULL_SCALE_RANGE_A
128     NY, OBGyroSampleRate sampleRate = OB_GYRO_SAMPLE_RATE_ANY) const {
129     ob_error *error = nullptr;
130     ob_config_enable_gyro_stream(impl_, fullScaleRange, sampleRate, &error);
131     Error::handle(&error);
132 }
133
134 void enableAllStream() {
135     ob_error *error = nullptr;
136     ob_config_enable_all_stream(impl_, &error);
137     Error::handle(&error);
138 }
139
140 void disableStream(OBStreamType streamType) const {
141     ob_error *error = nullptr;
142     ob_config_disable_stream(impl_, streamType, &error);
143     Error::handle(&error);
144 }
145
146 void disableStream(OBSensorType sensorType) const {
147     auto streamType = ob::TypeHelper::convertSensorTypeToStreamType(sensorType);
148     disableStream(streamType);
149 }
150
151 void disableAllStream() const {
152     ob_error *error = nullptr;
153     ob_config_disable_all_stream(impl_, &error);
154     Error::handle(&error);
155 }
156
157 std::shared_ptr<StreamProfileList> getEnabledStreamProfileList() const {
158     ob_error *error = nullptr;
159     auto list = ob_config_get_enabled_stream_profile_list(impl_, &error);
160     Error::handle(&error);
161     return std::make_shared<StreamProfileList>(list);
162 }
163
164 void setAlignMode(OBAAlignMode mode) const {
165     ob_error *error = nullptr;
166     ob_config_set_align_mode(impl_, mode, &error);
167     Error::handle(&error);
168 }
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222

```

```

223
229 void setDepthScaleRequire(bool enable) const {
230     ob_error *error = nullptr;
231     ob_config_set_depth_scale_after_align_require(impl_, enable, &error);
232     Error::handle(&error);
233 }
234
242 void setFrameAggregateOutputMode(OBFrameAggregateOutputMode mode) const {
243     ob_error *error = nullptr;
244     ob_config_set_frame_aggregate_output_mode(impl_, mode, &error);
245     Error::handle(&error);
246 }
247 };
248
249 class Pipeline {
250 public:
256     typedef std::function<void(std::shared_ptr<FrameSet> frame)> FrameSetCallback;
257
258 private:
259     ob_pipeline_t *impl_;
260     FrameSetCallback callback_;
261
262 public:
268     Pipeline() {
269         ob_error *error = nullptr;
270         impl_ = ob_create_pipeline(&error);
271         Error::handle(&error);
272     }
273
279     explicit Pipeline(std::shared_ptr<Device> device) {
280         ob_error *error = nullptr;
281         impl_ = ob_create_pipeline_with_device(device->getImpl(), &error);
282         Error::handle(&error);
283     }
284
288     ~Pipeline() noexcept {
289         ob_error *error = nullptr;
290         ob_delete_pipeline(impl_, &error);
291         Error::handle(&error, false);
292     }
293
299     void start(std::shared_ptr<Config> config = nullptr) const {
300         ob_error *error = nullptr;
301         ob_config_t *config_impl = config == nullptr ? nullptr : config->getImpl();
302         ob_pipeline_start_with_config(impl_, config_impl, &error);
303         Error::handle(&error);
304     }
305
312     void start(std::shared_ptr<Config> config, FrameSetCallback callback) {
313         callback_ = callback;
314         ob_error *error = nullptr;
315         ob_pipeline_start_with_callback(impl_, config ? config->getImpl() : nullptr, &Pipeline::frameSetCallback, this, &error);
316         Error::handle(&error);
317     }
318
319     static void frameSetCallback(ob_frame_t *frameSet, void *userData) {
320         auto pipeline = static_cast<Pipeline *>(userData);
321         pipeline->callback_(std::make_shared<FrameSet>(frameSet));
322     }
323
327     void stop() const {
328         ob_error *error = nullptr;
329         ob_pipeline_stop(impl_, &error);
330         Error::handle(&error);
331     }

```

```

351 }
352 std::shared_ptr<Config> getConfig() const {
353     ob_error *error = nullptr;
354     auto config = ob_pipeline_get_config(impl_, &error);
355     Error::handle(&error);
356     return std::make_shared<Config>(config);
357 }
358
359 std::shared_ptr<FrameSet> waitForFrameset(uint32_t timeoutMs = 1000) const {
360     ob_error *error = nullptr;
361     auto frameSet = ob_pipeline_wait_for_frameset(impl_, timeoutMs, &error);
362     if(frameSet == nullptr) {
363         return nullptr;
364     }
365     Error::handle(&error);
366     return std::make_shared<FrameSet>(frameSet);
367 }
368
369 std::shared_ptr<Device> getDevice() const {
370     ob_error *error = nullptr;
371     auto device = ob_pipeline_get_device(impl_, &error);
372     Error::handle(&error);
373     return std::make_shared<Device>(device);
374 }
375
376 std::shared_ptr<StreamProfileList> getStreamProfileList(OBSensorType sensorType) const {
377     ob_error *error = nullptr;
378     auto list = ob_pipeline_get_stream_profile_list(impl_, sensorType, &error);
379     Error::handle(&error);
380     return std::make_shared<StreamProfileList>(list);
381 }
382
383 std::shared_ptr<StreamProfileList> getD2CDepthProfileList(std::shared_ptr<StreamProfile> colorProfile,
384     OBAAlignMode alignMode) {
385     ob_error *error = nullptr;
386     auto list = ob_get_d2c_depth_profile_list(impl_, colorProfile->getImpl(), alignMode, &error);
387     Error::handle(&error);
388     return std::make_shared<StreamProfileList>(list);
389 }
390
391 void enableFrameSync() const {
392     ob_error *error = nullptr;
393     ob_pipeline_enable_frame_sync(impl_, &error);
394     Error::handle(&error);
395 }
396
397 void disableFrameSync() const {
398     ob_error *error = nullptr;
399     ob_pipeline_disable_frame_sync(impl_, &error);
400     Error::handle(&error);
401 }
402
403 public:
404     // The following interfaces are deprecated and are retained here for compatibility purposes.
405
406     OBCameraParam getCameraParam() {
407         ob_error *error = nullptr;
408         OBCameraParam cameraParam = ob_pipeline_get_camera_param(impl_, &error);
409         Error::handle(&error);
410         return cameraParam;
411     }
412
413     OBCameraParam getCameraParamWithProfile(uint32_t colorWidth, uint32_t colorHeight, uint32_t depth
414         Width, uint32_t depthHeight) {
415         ob_error *error = nullptr;

```

```
434     OBCameraParam cameraParam=ob_pipeline_get_camera_param_with_profile(impl_, colorWidth, colo
435     rHeight, depthWidth, depthHeight, &error);
436     Error::handle(&error);
437     return cameraParam;
438 }
439 OBCalibrationParam getCalibrationParam(std::shared_ptr<Config> config) {
440     ob_error *error = nullptr;
441     OBCalibrationParam calibrationParam=ob_pipeline_get_calibration_param(impl_, config->getImpl(),
442     &error);
443     Error::handle(&error);
444     return calibrationParam;
445 }
446 std::shared_ptr<FrameSet> waitForFrames(uint32_t timeoutMs = 1000) const {
447     return waitForFrameset(timeoutMs);
448 }
449 };
450
451 } // namespace ob
```

# Property.h File Reference

Control command property list maintenance. More...

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Classes

```
struct OBPropertyItem
```

Used to describe the characteristics of each property. More...

## Macros

```
#define OB_PROP_TIMER_RESET_TRIGGER_OUT_ENABLE_BOOL OB_PROP_TIMER_RESET_TRIGGER_OUT_ENABLE
#define OB_PROP_LASER_ON_OFF_MODE_INT OB_PROP_LASER_ON_OFF_PATTERN_INT
#define OB_PROP_LASER_ENERGY_LEVEL_INT OB_PROP_LASER_POWER_LEVEL_CONTROL
#define OB_PROP_LASER_HW_ENERGY_LEVEL_INT OB_PROP_LASER_POWER_ACTUAL_LEVEL
#define OB_PROP_DEVICE_USB3_REPEAT_IDENTITY_BOOL OB_PROP_DEVICE_USB2_REPEAT_IDENTITY
#define OB_PROP_DEPTH_SOFT_FILTER_BOOL OB_PROP_DEPTH_NOISE_REMOVAL_FILTER
#define OB_PROP_DEPTH_MAX_DIFF_INT OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX
#define OB_PROP_DEPTH_MAX_SPECKLE_SIZE_INT OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX
```

## Typedefs

```
typedef enum OBPropertyID ob_property_id
```

```
typedef enum OB.PropertyType OB.PropertyType
```

The data type used to describe all property settings.

```
typedef enum OB.PropertyType ob_property_type
```

```
typedef struct OBPropertyItem OBPropertyItem
```

Used to describe the characteristics of each property.

```
typedef struct OBPropertyItem ob_property_item
```

## Enumerations

```
enum OBPropertyID {
    OB_PROP_LDP_BOOL = 2, OB_PROP_LASER_BOOL = 3,
    OB_PROP_LASER_PULSE_WIDTH_INT = 4, OB_PROP_LASER_CURRENT_FLOAT = 5,
    OB_PROP_FLOOD_BOOL = 6, OB_PROP_FLOOD_LEVEL_INT = 7,
    OB_PROP_TEMPERATURE_COMPENSATION_BOOL = 8,
```

**OB\_PROP\_DEPTH\_MIRROR\_BOOL** = 14 ,  
**OB\_PROP\_DEPTH\_FLIP\_BOOL** = 15 , **OB\_PROP\_DEPTH\_POSTFILTER\_BOOL** = 16 ,  
**OB\_PROP\_DEPTH\_HOLEFILTER\_BOOL** = 17 , **OB\_PROP\_IR\_MIRROR\_BOOL** = 18 ,  
**OB\_PROP\_IR\_FLIP\_BOOL** = 19 , **OB\_PROP\_MIN\_DEPTH\_INT** = 22 ,  
**OB\_PROP\_MAX\_DEPTH\_INT** = 23 ,  
**OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_BOOL** = 24 ,  
**OB\_PROP\_LDP\_STATUS\_BOOL** = 32 ,  
**OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_MAX\_DIFF\_INT** = 40 ,  
**OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_MAX\_SPECKLE\_SIZE\_INT** = 41 ,  
**OB\_PROP\_DEPTH\_ALIGN\_HARDWARE\_BOOL** = 42 ,  
**OB\_PROP\_TIMESTAMP\_OFFSET\_INT** = 43 ,  
**OB\_PROP\_HARDWARE\_DISTORTION\_SWITCH\_BOOL** = 61 ,  
**OB\_PROP\_FAN\_WORK\_MODE\_INT** = 62 ,  
**OB\_PROP\_DEPTH\_ALIGN\_HARDWARE\_MODE\_INT** = 63 ,  
**OB\_PROP\_ANII\_COLLUSION\_ACTIVATION\_STATUS\_BOOL** = 64 ,  
**OB\_PROP\_DEPTH\_PRECISION\_LEVEL\_INT** = 75 , **OB\_PROP\_TOF\_FILTER\_RANGE\_INT** = 76 , **OB\_PROP\_LASER\_MODE\_INT** = 79 ,  
**OB\_PROP\_RECTIFY2\_BOOL** = 80 , **OB\_PROP\_COLOR\_MIRROR\_BOOL** = 81 ,  
**OB\_PROP\_COLOR\_FLIP\_BOOL** = 82 , **OB\_PROP\_INDICATOR\_LIGHT\_BOOL** = 83 ,  
**OB\_PROP\_DISPARITY\_TO\_DEPTH\_BOOL** = 85 , **OB\_PROP\_BRT\_BOOL** = 86 ,  
**OB\_PROP\_WATCHDOG\_BOOL** = 87 , **OB\_PROP\_EXTERNAL\_SIGNAL\_RESET\_BOOL** = 88 ,  
**OB\_PROP\_HEARTBEAT\_BOOL** = 89 , **OB\_PROP\_DEPTH\_CROPPING\_MODE\_INT** = 90 ,  
**OB\_PROP\_D2C\_PREPROCESS\_BOOL** = 91 , **OB\_PROP\_GPM\_BOOL** = 93 ,  
**OB\_PROP\_RGB\_CUSTOM\_CROP\_BOOL** = 94 , **OB\_PROP\_DEVICE\_WORK\_MODE\_INT** = 95 , **OB\_PROP\_DEVICE\_COMMUNICATION\_TYPE\_INT** = 97 ,  
**OB\_PROP\_SWITCH\_IR\_MODE\_INT** = 98 ,  
**OB\_PROP\_LASER\_POWER\_LEVEL\_CONTROL\_INT** = 99 ,  
**OB\_PROP\_LDP\_MEASURE\_DISTANCE\_INT** = 100 ,  
**OB\_PROP\_TIMER\_RESET\_SIGNAL\_BOOL** = 104 ,  
**OB\_PROP\_TIMER\_RESET\_TRIGGER\_OUT\_ENABLE\_BOOL** = 105 ,  
**OB\_PROP\_TIMER\_RESET\_DELAY\_US\_INT** = 106 ,  
**OB\_PROP\_CAPTURE\_IMAGE\_SIGNAL\_BOOL** = 107 ,  
**OB\_PROP\_IR\_RIGHT\_MIRROR\_BOOL** = 112 ,  
**OB\_PROP\_CAPTURE\_IMAGE\_FRAME\_NUMBER\_INT** = 113 ,  
**OB\_PROP\_IR\_RIGHT\_FLIP\_BOOL** = 114 , **OB\_PROP\_COLOR\_ROTATE\_INT** = 115 ,  
**OB\_PROP\_IR\_ROTATE\_INT** = 116 , **OB\_PROP\_IR\_RIGHT\_ROTATE\_INT** = 117 ,  
**OB\_PROP\_DEPTH\_ROTATE\_INT** = 118 ,  
**OB\_PROP\_LASER\_POWER\_ACTUAL\_LEVEL\_INT** = 119 ,  
**OB\_PROP\_USB\_POWER\_STATE\_INT** = 121 , **OB\_PROP\_DC\_POWER\_STATE\_INT** = 122 ,  
**OB\_PROP\_DEVICE\_DEVELOPMENT\_MODE\_INT** = 129 ,  
**OB\_PROP\_SYNC\_SIGNAL\_TRIGGER\_OUT\_BOOL** = 130 ,  
**OB\_PROP\_RESTORE\_FACTORY\_SETTINGS\_BOOL** = 131 ,  
**OB\_PROP\_BOOT\_INTO\_RECOVERY\_MODE\_BOOL** = 132 ,

**OB\_PROP\_DEVICE\_IN\_RECOVERY\_MODE\_BOOL** = 133 ,  
**OB\_PROP\_CAPTURE\_INTERVAL\_MODE\_INT** = 134 ,  
**OB\_PROP\_CAPTURE\_IMAGE\_TIME\_INTERVAL\_INT** = 135 ,  
**OB\_PROP\_CAPTURE\_IMAGE\_NUMBER\_INTERVAL\_INT** = 136 ,  
    **OB\_PROP\_TIMER\_RESET\_ENABLE\_BOOL** = 140 ,  
**OB\_PROP\_DEVICE\_USB2\_REPEAT\_IDENTIFY\_BOOL** = 141 ,  
**OB\_PROP\_DEVICE\_REBOOT\_DELAY\_INT** = 142 ,  
**OB\_PROP\_LASER\_OVERCURRENT\_PROTECTION\_STATUS\_BOOL** = 148 ,  
    **OB\_PROP\_LASER\_PULSE\_WIDTH\_PROTECTION\_STATUS\_BOOL** = 149 ,  
**OB\_PROP\_LASER\_ALWAYS\_ON\_BOOL** = 174 ,  
**OB\_PROP\_LASER\_ON\_OFF\_PATTERN\_INT** = 175 ,  
**OB\_PROP\_DEPTH\_UNIT\_FLEXIBLE\_ADJUSTMENT\_FLOAT** = 176 ,  
    **OB\_PROP\_LASER\_CONTROL\_INT** = 182 , **OB\_PROP\_IR\_BRIGHTNESS\_INT** = 184 ,  
**OB\_PROP\_SLAVE\_DEVICE\_SYNC\_STATUS\_BOOL** = 188 ,  
**OB\_PROP\_COLOR\_AE\_MAX\_EXPOSURE\_INT** = 189 ,  
    **OB\_PROP\_IR\_AE\_MAX\_EXPOSURE\_INT** = 190 ,  
**OB\_PROP\_DISP\_SEARCH\_RANGE\_MODE\_INT** = 191 ,  
**OB\_PROP\_LASER\_HIGH\_TEMPERATURE\_PROTECT\_BOOL** = 193 ,  
**OB\_PROP\_LOW\_EXPOSURE\_LASER\_CONTROL\_BOOL** = 194 ,  
    **OB\_PROP\_CHECK\_PPS\_SYNC\_IN\_SIGNAL\_BOOL** = 195 ,  
**OB\_PROP\_DISP\_SEARCH\_OFFSET\_INT** = 196 , **OB\_PROP\_DEVICE\_REPOWER\_BOOL** =  
202 , **OB\_PROP\_FRAME\_INTERLEAVE\_CONFIG\_INDEX\_INT** = 204 ,  
    **OB\_PROP\_FRAME\_INTERLEAVE\_ENABLE\_BOOL** = 205 ,  
**OB\_PROP\_FRAME\_INTERLEAVE\_LASER\_PATTERN\_SYNC\_DELAY\_INT** = 206 ,  
**OB\_PROP\_ON\_CHIP\_CALIBRATION\_HEALTH\_CHECK\_FLOAT** = 209 ,  
**OB\_PROP\_ON\_CHIP\_CALIBRATION\_ENABLE\_BOOL** = 210 ,  
    **OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_ENABLE\_BOOL** = 211 ,  
**OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_THRESHOLD\_FLOAT** = 212 ,  
**OB\_DEVICE\_AUTO\_CAPTURE\_ENABLE\_BOOL** = 216 ,  
**OB\_DEVICE\_AUTO\_CAPTURE\_INTERVAL\_TIME\_INT** = 217 ,  
    **OB\_DEVICE\_PTP\_CLOCK\_SYNC\_ENABLE\_BOOL** = 223 ,  
**OB\_PROP\_DEPTH\_WITH\_CONFIDENCE\_STREAM\_ENABLE\_BOOL** = 224 ,  
**OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_BOOL** = 226 ,  
**OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_THRESHOLD\_INT** = 227 ,  
    **OB\_PROP\_CONFIDENCE\_MIRROR\_BOOL** = 229 ,  
**OB\_PROP\_CONFIDENCE\_FLIP\_BOOL** = 230 , **OB\_PROP\_CONFIDENCE\_ROTATE\_INT** =  
231 , **OB\_STRUCT\_BASELINE\_CALIBRATION\_PARAM** = 1002 ,  
    **OB\_STRUCT\_DEVICE\_TEMPERATURE** = 1003 ,  
**OB\_STRUCT\_TOF\_EXPOSURE\_THRESHOLD\_CONTROL** = 1024 ,  
**OB\_STRUCT\_DEVICE\_SERIAL\_NUMBER** = 1035 , **OB\_STRUCT\_DEVICE\_TIME** = 1037 ,  
    **OB\_STRUCT\_MULTI\_DEVICE\_SYNC\_CONFIG** = 1038 , **OB\_STRUCT\_RGB\_CROP\_ROI**  
= 1040 , **OB\_STRUCT\_DEVICE\_IP\_ADDR\_CONFIG** = 1041 ,  
**OB\_STRUCT\_CURRENT\_DEPTH\_ALG\_MODE** = 1043 ,  
**OB\_STRUCT\_DEPTH\_PRECISION\_SUPPORT\_LIST** = 1045 ,

```
OB_STRUCT_DEVICE_STATIC_IP_CONFIG_RECORD = 1053 ,
OB_STRUCT_DEPTH_HDR_CONFIG = 1059 , OB_STRUCT_COLOR_AE_ROI = 1060 ,
OB_STRUCT_DEPTH_AE_ROI = 1061 , OB_STRUCT_ASIC_SERIAL_NUMBER = 1063 ,
OB_STRUCT_DISP_OFFSET_CONFIG = 1064 ,
OB_STRUCT_PRESET_RESOLUTION_CONFIG = 1069 ,
OB_PROP_COLOR_AUTO_EXPOSURE_BOOL = 2000 ,
OB_PROP_COLOR_EXPOSURE_INT = 2001 , OB_PROP_COLOR_GAIN_INT = 2002 ,
OB_PROP_COLOR_AUTO_WHITE_BALANCE_BOOL = 2003 ,
OB_PROP_COLOR_WHITE_BALANCE_INT = 2004 ,
OB_PROP_COLOR_BRIGHTNESS_INT = 2005 , OB_PROP_COLOR_SHARPNESS_INT =
2006 , OB_PROP_COLOR_SHUTTER_INT = 2007 ,
OB_PROP_COLOR_SATURATION_INT = 2008 , OB_PROP_COLOR_CONTRAST_INT =
2009 , OB_PROP_COLOR_GAMMA_INT = 2010 , OB_PROP_COLOR_ROLL_INT = 2011 ,
OB_PROP_COLOR_AUTO_EXPOSURE_PRIORITY_INT = 2012 ,
OB_PROP_COLOR_BACKLIGHT_COMPENSATION_INT = 2013 ,
OB_PROP_COLOR_HUE_INT = 2014 ,
OB_PROP_COLOR_POWER_LINE_FREQUENCY_INT = 2015 ,
OB_PROP_DEPTH_AUTO_EXPOSURE_BOOL = 2016 ,
OB_PROP_DEPTH_EXPOSURE_INT = 2017 , OB_PROP_DEPTH_GAIN_INT = 2018 ,
OB_PROP_IR_AUTO_EXPOSURE_BOOL = 2025 ,
OB_PROP_IR_EXPOSURE_INT = 2026 , OB_PROP_IR_GAIN_INT = 2027 ,
OB_PROP_IR_CHANNEL_DATA_SOURCE_INT = 2028 ,
OB_PROP_DEPTH_RM_FILTER_BOOL = 2029 ,
OB_PROP_COLOR_MAXIMAL_GAIN_INT = 2030 ,
OB_PROP_COLOR_MAXIMAL_SHUTTER_INT = 2031 ,
OB_PROP_IR_SHORT_EXPOSURE_BOOL = 2032 , OB_PROP_COLOR_HDR_BOOL =
2034 ,
OB_PROP_IR_LONG_EXPOSURE_BOOL = 2035 , OB_PROP_SKIP_FRAME_BOOL =
2036 , OB_PROP_HDR_MERGE_BOOL = 2037 , OB_PROP_COLOR_FOCUS_INT = 2038 ,
OB_PROP_IR_RECTIFY_BOOL = 2040 ,
OB_PROP_DEPTH_AUTO_EXPOSURE_PRIORITY_INT = 2052 ,
OB_PROP_SDK_DISPARITY_TO_DEPTH_BOOL = 3004 ,
OB_PROP_SDK_DEPTH_FRAME_UNPACK_BOOL = 3007 ,
OB_PROP_SDK_IR_FRAME_UNPACK_BOOL = 3008 ,
OB_PROP_SDK_ACCEL_FRAME_TRANSFORMED_BOOL = 3009 ,
OB_PROP_SDK_GYRO_FRAME_TRANSFORMED_BOOL = 3010 ,
OB_PROP_SDK_IR_LEFT_FRAME_UNPACK_BOOL = 3011 ,
OB_PROP_SDK_IR_RIGHT_FRAME_UNPACK_BOOL = 3012 ,
OB_PROP_NETWORK_BANDWIDTH_TYPE_INT = 3027 ,
OB_PROP_DEVICE_PERFORMANCE_MODE_INT = 3028 ,
OB_RAW_DATA_CAMERA_CALIB_JSON_FILE = 4029 ,
OB_PROP_DEBUG_ESGM_CONFIDENCE_FLOAT = 5013
}
```

Enumeration value describing all attribute control commands of the device. [More...](#)

```
enum OBPropertyType { OB_BOOL_PROPERTY = 0 , OB_INT_PROPERTY = 1 ,  
OB_FLOAT_PROPERTY = 2 , OB_STRUCT_PROPERTY = 3 }
```

The data type used to describe all property settings. [More...](#)

## Detailed Description

Control command property list maintenance.

Definition in file [Property.h](#).

## Macro Definition Documentation

### ◆ OB\_PROP\_TIMER\_RESET\_TRIGGER\_OUT\_ENABLE\_BOOL

```
#define  
OB_PROP_TIMER_RESET_TRIGGER_OUT_ENABLE_BOOL OB_PROP_TIMER_RESET_TRIGGER_OUT_ENAB
```

Definition at line [870](#) of file [Property.h](#).

### ◆ OB\_PROP\_LASER\_ON\_OFF\_MODE\_INT

```
#define OB_PROP_LASER_ON_OFF_MODE_INT OB_PROP_LASER_ON_OFF_PATTERN_INT
```

Definition at line [871](#) of file [Property.h](#).

### ◆ OB\_PROP\_LASER\_ENERGY\_LEVEL\_INT

```
#define OB_PROP_LASER_ENERGY_LEVEL_INT OB_PROP_LASER_POWER_LEVEL_CONTROL_INT
```

Definition at line [872](#) of file [Property.h](#).

### ◆ OB\_PROP\_LASER\_HW\_ENERGY\_LEVEL\_INT

```
#define OB_PROP_LASER_HW_ENERGY_LEVEL_INT OB_PROP_LASER_POWER_ACTUAL_LEVEL_INT
```

Definition at line [873](#) of file [Property.h](#).

◆ OB\_PROP\_DEVICE\_USB3\_REPEAT\_IDENTIFY\_BOOL

```
#define  
OB_PROP_DEVICE_USB3_REPEAT_IDENTIFY_BOOL OB_PROP_DEVICE_USB2_REPEAT_IDENTIFY_BOOL
```

Definition at line [874](#) of file [Property.h](#).

◆ OB\_PROP\_DEPTH\_SOFT\_FILTER\_BOOL

```
#define OB_PROP_DEPTH_SOFT_FILTER_BOOL OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_BOOL
```

Definition at line [875](#) of file [Property.h](#).

◆ OB\_PROP\_DEPTH\_MAX\_DIFF\_INT

```
#define  
OB_PROP_DEPTH_MAX_DIFF_INT OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_DIFF_INT
```

Definition at line [876](#) of file [Property.h](#).

◆ OB\_PROP\_DEPTH\_MAX\_SPECKLE\_SIZE\_INT

```
#define  
OB_PROP_DEPTH_MAX_SPECKLE_SIZE_INT OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_SPECK
```

Definition at line [877](#) of file [Property.h](#).

## Typedef Documentation

◆ ob\_property\_id

```
typedef enum OBPropertyID ob_property_id
```

◆ OBPropertyType

```
typedef enum OBPropertyType OBPropertyType
```

The data type used to describe all property settings.

◆ **ob\_property\_type**

```
typedef enum OBPropertyType ob_property_type
```

◆ **OBPropertyItem**

```
typedef struct OBPropertyItem OBPropertyItem
```

Used to describe the characteristics of each property.

◆ **ob\_property\_item**

```
typedef struct OBPropertyItem ob_property_item
```

## Enumeration Type Documentation

◆ **OBPropertyID**

```
enum OBPropertyID
```

Enumeration value describing all attribute control commands of the device.

Enumerator

OB_PROP_LDP_BOOL	LDP switch.
OB_PROP LASER_BOOL	Laser switch.
OB_PROP LASER_PULSE_WIDTH_INT	laser pulse width
OB_PROP LASER_CURRENT_FLOAT	Laser current (uint: mA)
OB_PROP_FLOOD_BOOL	IR flood switch.
OB_PROP_FLOOD_LEVEL_INT	IR flood level.
OB_PROP_TEMPERATURE_COMPENSATION_BOOL	Enable/disable temperature
OB_PROP_DEPTH_MIRROR_BOOL	Depth mirror.

OB_PROP_DEPTH_FLIP_BOOL	Depth flip.
OB_PROP_DEPTH_POSTFILTER_BOOL	Depth Postfilter.
OB_PROP_DEPTH_HOLEFILTER_BOOL	Depth Holefilter.
OB_PROP_IR_MIRROR_BOOL	IR mirror.
OB_PROP_IR_FLIP_BOOL	IR flip.
OB_PROP_MIN_DEPTH_INT	Minimum depth threshold.
OB_PROP_MAX_DEPTH_INT	Maximum depth threshold
OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_BOOL	Software filter switch.
OB_PROP_LDP_STATUS_BOOL	LDP status.
OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_DIFF_INT	maxdiff for depth noise re
OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_SPECKLE_SIZE_INT	maxSpeckleSize for depth
OB_PROP_DEPTH_ALIGN_HARDWARE_BOOL	Hardware d2c is on.
OB_PROP_TIMESTAMP_OFFSET_INT	Timestamp adjustment.
OB_PROP_HARDWARE_DISTORTION_SWITCH_BOOL	Hardware distortion switc
OB_PROP_FAN_WORK_MODE_INT	Fan mode switch.
OB_PROP_DEPTH_ALIGN_HARDWARE_MODE_INT	Multi-resolution D2C mod
OB_PROP_ANTI_COLLUSION_ACTIVATION_STATUS_BOOL	Anti_collusion activation s
OB_PROP_DEPTH_PRECISION_LEVEL_INT	the depth precision level, v unit, needs to be confirme DepthFrame
OB_PROP_TOF_FILTER_RANGE_INT	tof filter range configurati
OB_PROP_LASER_MODE_INT	laser mode, the firmware 1 2: Torch
OB_PROP_RECTIFY2_BOOL	brt2r-rectify function swit 0: Disable, 1: Rectify Ena
OB_PROP_COLOR_MIRROR_BOOL	Color mirror.
OB_PROP_COLOR_FLIP_BOOL	Color flip.
OB_PROP_INDICATOR_LIGHT_BOOL	Indicator switch, 0: Disab
OB_PROP_DISPARITY_TO_DEPTH_BOOL	Disparity to depth switch, convert to depth, true: sw depth.
OB_PROP_BRT_BOOL	BRT function switch (anti Enable.
OB_PROP_WATCHDOG_BOOL	Watchdog function switch
OB_PROP_EXTERNAL_SIGNAL_RESET_BOOL	External signal trigger rest

OB_PROP_HEARTBEAT_BOOL	Heartbeat monitoring func
OB_PROP_DEPTH_CROPPING_MODE_INT	Depth cropping mode dev
OB_PROP_D2C_PREPROCESS_BOOL	D2C preprocessing switch
OB_PROP_GPM_BOOL	Enable/disable GPM funct
OB_PROP_RGB_CUSTOM_CROP_BOOL	Custom RGB cropping sw and the ROI cropping area.
OB_PROP_DEVICE_WORK_MODE_INT	Device operating mode (power).
OB_PROP_DEVICE_COMMUNICATION_TYPE_INT	Device communication type.
OB_PROP_SWITCH_IR_MODE_INT	Switch infrared imaging mode.
OB_PROP_LASER_POWER_LEVEL_CONTROL_INT	Laser power level.
OB_PROP_LDP_MEASURE_DISTANCE_INT	LDP's measure distance, unit mm.
OB_PROP_TIMER_RESET_SIGNAL_BOOL	Reset device time to zero.
OB_PROP_TIMER_RESET_TRIGGER_OUT_ENABLE_BOOL	Enable send reset device to false: disable.
OB_PROP_TIMER_RESET_DELAY_US_INT	Delay to reset device time.
OB_PROP_CAPTURE_IMAGE_SIGNAL_BOOL	Signal to capture image.
OB_PROP_IR_RIGHT_MIRROR_BOOL	Right IR sensor mirror state.
OB_PROP_CAPTURE_IMAGE_FRAME_NUMBER_INT	Number frame to capture [0~255].
OB_PROP_IR_RIGHT_FLIP_BOOL	Right IR sensor flip state. false.
OB_PROP_COLOR_ROTATE_INT	Color sensor rotation, angle.
OB_PROP_IR_ROTATE_INT	IR/Left-IR sensor rotation.
OB_PROP_IR_RIGHT_ROTATE_INT	Right IR sensor rotation, angle.
OB_PROP_DEPTH_ROTATE_INT	Depth sensor rotation, angle.
OB_PROP_LASER_POWER_ACTUAL_LEVEL_INT	Get hardware laser power element. OB_PROP_LASER_POWER will effect this command to laser energy level.
OB_PROP_USB_POWER_STATE_INT	USB's power state, enum type.
OB_PROP_DC_POWER_STATE_INT	DC's power state, enum type.

OB_PROP_DEVICE_DEVELOPMENT_MODE_INT	Device development mode definition in <a href="#">OBDeviceDef</a> <b>OB_USER_MODE</b>
	<b>Attention</b> The device takes modes.
OB_PROP_SYNC_SIGNAL_TRIGGER_OUT_BOOL	Multi-DeviceSync synchronization true: enable, false: disable.
OB_PROP_RESTORE_FACTORY_SETTINGS_BOOL	Restore factory settings at
	<b>Attention</b> This command changes the value must be true before restarting the dev
OB_PROP_BOOT_INTO_RECOVERY_MODE_BOOL	Enter recovery mode (flas
	<b>Attention</b> The device will take this option. After entering the device system, it may cause system damage, please be careful.
OB_PROP_DEVICE_IN_RECOVERY_MODE_BOOL	Query whether the current device is in recovery mode (read-only)
OB_PROP_CAPTURE_INTERVAL_MODE_INT	Capture interval mode, 0:t
OB_PROP_CAPTURE_IMAGE_TIME_INTERVAL_INT	Capture time interval.
OB_PROP_CAPTURE_IMAGE_NUMBER_INTERVAL_INT	Capture number interval.
OB_PROP_TIMER_RESET_ENABLE_BOOL	
OB_PROP_DEVICE_USB2_REPEAT_IDENTIFY_BOOL	Enable or disable the device's self-identification when the device is connected to a computer.
	This feature ensures that the device can be recognized by a USB 2.0 device when connected.
OB_PROP_DEVICE_REBOOT_DELAY_INT	Reboot device delay mode
OB_PROP_LASER_OVERCURRENT_PROTECTION_STATUS_BOOL	Query the status of laser current protection
OB_PROP_LASER_PULSE_WIDTH_PROTECTION_STATUS_BOOL	Query the status of laser pulse width protection

OB_PROP_LASER_ALWAYS_ON_BOOL	Laser always on, true: always on when out of exposure time.
OB_PROP_LASER_ON_OFF_PATTERN_INT	Laser on/off alternate mode: laser alternate.
<b>Attention</b>	
	When turn on this property, laser will turn off alternately each second.
OB_PROP_DEPTH_UNIT_FLEXIBLE_ADJUSTMENT_FLOAT	Depth unit flexible adjustment.
	This property allows control depth unit's fixed value.
<b>OB_PROP_DEPTH_PRESET_FLOAT</b>	some fixed value.
OB_PROP_LASER_CONTROL_INT	Laser control, 0: off, 1: on.
OB_PROP_IR_BRIGHTNESS_INT	IR brightness.
OB_PROP_SLAVE_DEVICE_SYNC_STATUS_BOOL	Slave/secondary device sync status.
OB_PROP_COLOR_AE_MAX_EXPOSURE_INT	Color AE max exposure.
OB_PROP_IR_AE_MAX_EXPOSURE_INT	Max exposure time of IR AE.
OB_PROP_DISP_SEARCH_RANGE_MODE_INT	Disparity search range mode.
OB_PROP_LASER_HIGH_TEMPERATURE_PROTECT_BOOL	Laser high temperature protection.
OB_PROP_LOW_EXPOSURE_LASER_CONTROL_BOOL	low exposure laser control.
	Currently using for Dabai camera. When the temperature is higher than a certain threshold, the laser is turned off. When the temperature is lower than the threshold, the laser is turned on.
OB_PROP_CHECK_PPS_SYNC_IN_SIGNAL_BOOL	check pps sync in signal.
OB_PROP_DISP_SEARCH_OFFSET_INT	Disparity search range offset.
OB_PROP_DEVICE_REPOWER_BOOL	Repower device (cut off power).
	Currently using for GMSI camera. When the host drive sends a command to enable frame interleaving, the camera will enable frame interleaving configuration. When the host drive sends a command to disable frame interleaving, the camera will disable frame interleaving configuration.
OB_PROP_FRAME_INTERLEAVE_CONFIG_INDEX_INT	frame interleave config index.
OB_PROP_FRAME_INTERLEAVE_ENABLE_BOOL	frame interleave enable (true/false).
OB_PROP_FRAME_INTERLEAVE_LASER_PATTERN_SYNC_DELAY_INT	laser pattern sync with delay.
OB_PROP_ON_CHIP_CALIBRATION_HEALTH_CHECK_FLOAT	Get the health check result.
OB_PROP_ON_CHIP_CALIBRATION_ENABLE_BOOL	Enable or disable on-chip calibration.
OB_PROP_HW_NOISE_REMOVE_FILTER_ENABLE_BOOL	hardware noise remove filter enable.

OB_PROP_HW_NOISE_REMOVE_FILTER_THRESHOLD_FLOAT	hardware noise remove fil
OB_DEVICE_AUTO_CAPTURE_ENABLE_BOOL	soft trigger auto capture e
OB_DEVICE_AUTO_CAPTURE_INTERVAL_TIME_INT	OB_MULTI_DEVICE_SY mode
OB_DEVICE_PTP_CLOCK_SYNC_ENABLE_BOOL	soft trigger auto capture i
OB_PROP_DEPTH_WITH_CONFIDENCE_STREAM_ENABLE_BOOL	OB_MULTI_DEVICE_SY
OB_PROP_CONFIDENCE_STREAM_FILTER_BOOL	mode
OB_PROP_CONFIDENCE_STREAM_FILTER_THRESHOLD_INT	PTP time synchronization
OB_PROP_CONFIDENCE_MIRROR_BOOL	Depth with confidence str
OB_PROP_CONFIDENCE_FLIP_BOOL	Enable or disable confiden
OB_PROP_CONFIDENCE_ROTATE_INT	Confidence stream filter t
OB_STRUCT_BASELINE_CALIBRATION_PARAM	Confidence stream mirror
OB_STRUCT_DEVICE_TEMPERATURE	Confidence stream flip en:
OB_STRUCT_TOF_EXPOSURE_THRESHOLD_CONTROL	Confidence stream rotate :
OB_STRUCT_DEVICE_SERIAL_NUMBER	Baseline calibration param
OB_STRUCT_DEVICE_TIME	Device temperature inform
OB_STRUCT_MULTI_DEVICE_SYNC_CONFIG	TOF exposure threshold r
OB_STRUCT_RGB_CROP_ROI	get/set serial number
OB_STRUCT_DEVICE_IP_ADDR_CONFIG	get/set device time
OB_STRUCT_CURRENT_DEPTH_ALG_MODE	Multi-device synchronizat
OB_STRUCT_DEPTH_PRECISION_SUPPORT_LIST	RGB cropping ROI.
OB_STRUCT_DEVICE_STATIC_IP_CONFIG_RECORD	Device IP address config
	The current camera depth
	A list of depth accuracy le
	corresponding to the enum
	Device network static ip c
	Using for get last static ip
	when user set static ip co

### Attention

read only

OB_STRUCT_DEPTH_HDR_CONFIG	Using to configure the depth sensor's HDR configuration.
	The Value type is <b>OBHdr</b> .
	<b>Attention</b>
	After enable HDR, the color sensor's AE ROI will be disabled.
OB_STRUCT_COLOR_AE_ROI	Color Sensor AE ROI configuration.
	The Value type is <b>OBReg</b> .
OB_STRUCT_DEPTH_AE_ROI	Depth Sensor AE ROI configuration.
	The Value type is <b>OBReg</b> .
OB_STRUCT ASIC_SERIAL_NUMBER	Since the ir sensor is the same as the color sensor, this property will also affect the ir sensor's ASIC serial number.
OB_STRUCT_DISP_OFFSET_CONFIG	Disparity offset interleaving configuration.
OB_STRUCT_PRESET_RESOLUTION_CONFIG	Preset resolution ratio configuration.
OB_PROP_COLOR_AUTO_EXPOSURE_BOOL	Color camera auto exposure adjustment.
OB_PROP_COLOR_EXPOSURE_INT	Color camera exposure adjustment.
OB_PROP_COLOR_GAIN_INT	Color camera gain adjustment.
OB_PROP_COLOR_AUTO_WHITE_BALANCE_BOOL	Color camera automatic white balance adjustment.
OB_PROP_COLOR_WHITE_BALANCE_INT	Color camera white balance adjustment.
OB_PROP_COLOR_BRIGHTNESS_INT	Color camera brightness adjustment.
OB_PROP_COLOR_SHARPNESS_INT	Color camera sharpness adjustment.
OB_PROP_COLOR_SHUTTER_INT	Color camera shutter adjustment.
OB_PROP_COLOR_SATURATION_INT	Color camera saturation adjustment.
OB_PROP_COLOR_CONTRAST_INT	Color camera contrast adjustment.
OB_PROP_COLOR_GAMMA_INT	Color camera gamma adjustment.
OB_PROP_COLOR_ROLL_INT	Color camera image rotation adjustment.
OB_PROP_COLOR_AUTO_EXPOSURE_PRIORITY_INT	Color camera auto exposure priority.
OB_PROP_COLOR_BACKLIGHT_COMPENSATION_INT	Color camera brightness compensation.
OB_PROP_COLOR_HUE_INT	Color camera color tint.
OB_PROP_COLOR_POWER_LINE_FREQUENCY_INT	Color Camera Power Line Frequency.
OB_PROP_DEPTH_AUTO_EXPOSURE_BOOL	Automatic exposure of depth sensor synchronously under some conditions.

OB_PROP_DEPTH_EXPOSURE_INT	Depth camera exposure ac synchronously under som
OB_PROP_DEPTH_GAIN_INT	Depth camera gain adjustr synchronously under som
OB_PROP_IR_AUTO_EXPOSURE_BOOL	Infrared camera auto expc synchronously under som
OB_PROP_IR_EXPOSURE_INT	Infrared camera exposure set the depth camera sync
OB_PROP_IR_GAIN_INT	Infrared camera gain adjus synchronously under som
OB_PROP_IR_CHANNEL_DATA_SOURCE_INT	Select Infrared camera da exception. 0 : IR stream fi Right sensor;.
OB_PROP_DEPTH_RM_FILTER_BOOL	Depth effect dedistortion, with D2C function, RM_F D2C is enabled.
OB_PROP_COLOR_MAXIMAL_GAIN_INT	Color camera maximal gai
OB_PROP_COLOR_MAXIMAL_SHUTTER_INT	Color camera shutter gain.
OB_PROP_IR_SHORT_EXPOSURE_BOOL	The enable/disable switch only by a few devices.
OB_PROP_COLOR_HDR_BOOL	Color camera HDR.
OB_PROP_IR_LONG_EXPOSURE_BOOL	IR long exposure mode sv
OB_PROP_SKIP_FRAME_BOOL	Setting and getting the US true: frame skipping mode
OB_PROP_HDR_MERGE_BOOL	Depth HDR merge, true: c
OB_PROP_COLOR_FOCUS_INT	Color camera FOCUS.
OB_PROP_IR_RECTIFY_BOOL	ir rectify status,true: ir rec
OB_PROP_DEPTH_AUTO_EXPOSURE_PRIORITY_INT	Depth camera priority.
OB_PROP_SDK_DISPARITY_TO_DEPTH_BOOL	Software disparity to dept
OB_PROP_SDK_DEPTH_FRAME_UNPACK_BOOL	Depth data unpacking fun turned on by default, supp
OB_PROP_SDK_IR_FRAME_UNPACK_BOOL	IR data unpacking functio by default, support RLE/Y
OB_PROP_SDK_ACCEL_FRAME_TRANSFORMED_BOOL	Accel data conversion fun
OB_PROP_SDK_GYRO_FRAME_TRANSFORMED_BOOL	Gyro data conversion fun
OB_PROP_SDK_IR_LEFT_FRAME_UNPACK_BOOL	Left IR frame data unpack turned on by default, supp

OB_PROP_SDK_IR_RIGHT_FRAME_UNPACK_BOOL	Right IR frame data unpack be turned on by default, si
OB_PROP_NETWORK_BANDWIDTH_TYPE_INT	Read the current network whether it is Gigabit Ether
OB_PROP_DEVICE_PERFORMANCE_MODE_INT	Switch device performanc Mode and High Performar
OB_RAW_DATA_CAMERA_CALIB_JSON_FILE	Calibration JSON file read
OB_PROP_DEBUG_ESGM_CONFIDENCE_FLOAT	Confidence degree.

Definition at line **20** of file [Property.h](#).

## ◆ OB.PropertyType

### enum [OB.PropertyType](#)

The data type used to describe all property settings.

Enumerator

OB_BOOL_PROPERTY	Boolean property
OB_INT_PROPERTY	Integer property
OB_FLOAT_PROPERTY	Floating-point property
OB_STRUCT_PROPERTY	Struct property

Definition at line **882** of file [Property.h](#).

# Property.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8
9 #pragma once
10
11 #include "ObTypes.h"
12
13 #ifndef __cplusplus
14 extern "C" {
15 #endif
16
20 typedef enum {
24     OB_PROP_LDP_BOOL = 2,
25
29     OB_PROP_LASER_BOOL = 3,
30
34     OB_PROP_LASER_PULSE_WIDTH_INT = 4,
35
39     OB_PROP_LASER_CURRENT_FLOAT = 5,
40
44     OB_PROP_FLOOD_BOOL = 6,
45
49     OB_PROP_FLOOD_LEVEL_INT = 7,
50
55     OB_PROP_TEMPERATURE_COMPENSATION_BOOL = 8,
56
60     OB_PROP_DEPTH_MIRROR_BOOL = 14,
61
65     OB_PROP_DEPTH_FLIP_BOOL = 15,
66
70     OB_PROP_DEPTH_POSTFILTER_BOOL = 16,
71
75     OB_PROP_DEPTH_HOLEFILTER_BOOL = 17,
76
80     OB_PROP_IR_MIRROR_BOOL = 18,
81
85     OB_PROP_IR_FLIP_BOOL = 19,
86
90     OB_PROP_MIN_DEPTH_INT = 22,
91
95     OB_PROP_MAX_DEPTH_INT = 23,
96
100    OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_BOOL = 24,
101
105    OB_PROP_LDP_STATUS_BOOL = 32,
106
110    OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_DIFF_INT = 40,
111
115    OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_SPECKLE_SIZE_INT = 41,
116
120    OB_PROP_DEPTH_ALIGN_HARDWARE_BOOL = 42,
121
125    OB_PROP_TIMESTAMP_OFFSET_INT = 43,
126
130    OB_PROP_HARDWARE_DISTORTION_SWITCH_BOOL = 61,
131
135    OB_PROP_FAN_WORK_MODE_INT = 62,
```

```
136      - - - - -
140      OB_PROP_DEPTH_ALIGN_HARDWARE_MODE_INT = 63,
141
145      OB_PROP_ANTI_COLLUSION_ACTIVATION_STATUS_BOOL = 64,
146
151      OB_PROP_DEPTH_PRECISION_LEVEL_INT = 75,
152
156      OB_PROP_TOF_FILTER_RANGE_INT = 76,
157
161      OB_PROP_LASER_MODE_INT = 79,
162
166      OB_PROP_RECTIFY2_BOOL = 80,
167
171      OB_PROP_COLOR_MIRROR_BOOL = 81,
172
176      OB_PROP_COLOR_FLIP_BOOL = 82,
177
181      OB_PROP_INDICATOR_LIGHT_BOOL = 83,
182
186      OB_PROP_DISPARITY_TO_DEPTH_BOOL = 85,
187
191      OB_PROP_BRT_BOOL = 86,
192
196      OB_PROP_WATCHDOG_BOOL = 87,
197
201      OB_PROP_EXTERNAL_SIGNAL_RESET_BOOL = 88,
202
206      OB_PROP_HEARTBEAT_BOOL = 89,
207
211      OB_PROP_DEPTH_CROPPING_MODE_INT = 90,
212
216      OB_PROP_D2C_PREPROCESS_BOOL = 91,
217
221      OB_PROP_GPM_BOOL = 93,
222
226      OB_PROP_RGB_CUSTOM_CROP_BOOL = 94,
227
231      OB_PROP_DEVICE_WORK_MODE_INT = 95,
232
236      OB_PROP_DEVICE_COMMUNICATION_TYPE_INT = 97,
237
241      OB_PROP_SWITCH_IR_MODE_INT = 98,
242
246      OB_PROP_LASER_POWER_LEVEL_CONTROL_INT = 99,
247
251      OB_PROP_LDP_MEASURE_DISTANCE_INT = 100,
252
256      OB_PROP_TIMER_RESET_SIGNAL_BOOL = 104,
257
261      OB_PROP_TIMER_RESET_TRIGGER_OUT_ENABLE_BOOL = 105,
262
266      OB_PROP_TIMER_RESET_DELAY_US_INT = 106,
267
271      OB_PROP_CAPTURE_IMAGE_SIGNAL_BOOL = 107,
272
276      OB_PROP_IR_RIGHT_MIRROR_BOOL = 112,
277
281      OB_PROP_CAPTURE_IMAGE_FRAME_NUMBER_INT = 113,
282
286      OB_PROP_IR_RIGHT_FLIP_BOOL = 114,
287
291      OB_PROP_COLOR_ROTATE_INT = 115,
292
296      OB_PROP_IR_ROTATE_INT = 116,
297
```

```
301 OB_PROP_IR_RIGHT_ROTATE_INT = 117,  
302  
306 OB_PROP_DEPTH_ROTATE_INT = 118,  
307  
312 OB_PROP_LASER_POWER_ACTUAL_LEVEL_INT = 119,  
313  
317 OB_PROP_USB_POWER_STATE_INT = 121,  
318  
322 OB_PROP_DC_POWER_STATE_INT = 122,  
323  
329 OB_PROP_DEVICE DEVELOPMENT_MODE_INT = 129,  
330  
334 OB_PROP_SYNC_SIGNAL_TRIGGER_OUT_BOOL = 130,  
335  
340 OB_PROP_RESTORE_FACTORY_SETTINGS_BOOL = 131,  
341  
347 OB_PROP_BOOT_INTO_RECOVERY_MODE_BOOL = 132,  
348  
352 OB_PROP_DEVICE_IN_RECOVERY_MODE_BOOL = 133,  
353  
357 OB_PROP_CAPTURE_INTERVAL_MODE_INT = 134,  
358  
362 OB_PROP_CAPTURE_IMAGE_TIME_INTERVAL_INT = 135,  
363  
367 OB_PROP_CAPTURE_IMAGE_NUMBER_INTERVAL_INT = 136,  
368  
369 /*  
370 * @brief Timer reset function enable  
371 */  
372 OB_PROP_TIMER_RESET_ENABLE_BOOL = 140,  
373  
378 OB_PROP_DEVICE_USB2_REPEAT_IDENTIFY_BOOL = 141,  
379  
383 OB_PROP_DEVICE_REBOOT_DELAY_INT = 142,  
384  
388 OB_PROP_LASER_OVERCURRENT_PROTECTION_STATUS_BOOL = 148,  
389  
393 OB_PROP_LASER_PULSE_WIDTH_PROTECTION_STATUS_BOOL = 149,  
394  
398 OB_PROP_LASER_ALWAYS_ON_BOOL = 174,  
399  
404 OB_PROP_LASER_ON_OFF_PATTERN_INT = 175,  
405  
410 OB_PROP_DEPTH_UNIT_FLEXIBLE_ADJUSTMENT_FLOAT = 176,  
411  
416 OB_PROP_LASER_CONTROL_INT = 182,  
417  
421 OB_PROP_IR_BRIGHTNESS_INT = 184,  
422  
426 OB_PROP_SLAVE_DEVICE_SYNC_STATUS_BOOL = 188,  
427  
431 OB_PROP_COLOR_AE_MAX_EXPOSURE_INT = 189,  
432  
436 OB_PROP_IR_AE_MAX_EXPOSURE_INT = 190,  
437  
441 OB_PROP_DISP_SEARCH_RANGE_MODE_INT = 191,  
442  
446 OB_PROP_LASER_HIGH_TEMPERATURE_PROTECT_BOOL = 193,  
447  
454 OB_PROP_LOW_EXPOSURE_LASER_CONTROL_BOOL = 194,  
455  
459 OB_PROP_CHECK_PPS_SYNC_IN_SIGNAL_BOOL = 195,  
460  
464 OB_PROP_DISP_SEARCH_OFFSET_INT = 196,  
465  
471 OB_PROP_DEVICE_BEDPOWER_BOOL = 200
```

471 OB\_PROP\_DEVICE\_REFPOWER\_DUAL = 202,  
472  
473 OB\_PROP\_FRAME\_INTERLEAVE\_CONFIG\_INDEX\_INT = 204,  
474  
475 OB\_PROP\_FRAME\_INTERLEAVE\_ENABLE\_BOOL = 205,  
476 OB\_PROP\_FRAME\_INTERLEAVE\_LASER\_PATTERN\_SYNC\_DELAY\_INT = 206,  
477 OB\_PROP\_ON\_CHIP\_CALIBRATION\_HEALTH\_CHECK\_FLOAT = 209,  
478  
479 OB\_PROP\_ON\_CHIP\_CALIBRATION\_ENABLE\_BOOL = 210,  
480  
481 OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_ENABLE\_BOOL = 211,  
482 OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_THRESHOLD\_FLOAT = 212,  
483  
484 OB\_DEVICE\_AUTO\_CAPTURE\_ENABLE\_BOOL = 216,  
485 OB\_DEVICE\_AUTO\_CAPTURE\_INTERVAL\_TIME\_INT = 217,  
486  
487 OB\_DEVICE\_PTP\_CLOCK\_SYNC\_ENABLE\_BOOL = 223,  
488  
489 OB\_PROP\_DEPTH\_WITH\_CONFIDENCE\_STREAM\_ENABLE\_BOOL = 224,  
490  
491 OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_BOOL = 226,  
492  
493 OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_THRESHOLD\_INT = 227,  
494  
495 OB\_PROP\_CONFIDENCE\_MIRROR\_BOOL = 229,  
496  
497 OB\_PROP\_CONFIDENCE\_FLIP\_BOOL = 230,  
498  
499 OB\_PROP\_CONFIDENCE\_ROTATE\_INT = 231,  
500  
501 OB\_STRUCT\_BASELINE\_CALIBRATION\_PARAM = 1002,  
502  
503 OB\_STRUCT\_DEVICE\_TEMPERATURE = 1003,  
504  
505 OB\_STRUCT\_TOF\_EXPOSURE\_THRESHOLD\_CONTROL = 1024,  
506  
507 OB\_STRUCT\_DEVICE\_SERIAL\_NUMBER = 1035,  
508  
509 OB\_STRUCT\_DEVICE\_TIME = 1037,  
510  
511 OB\_STRUCT\_MULTI\_DEVICE\_SYNC\_CONFIG = 1038,  
512  
513 OB\_STRUCT\_RGB\_CROP\_ROI = 1040,  
514  
515 OB\_STRUCT\_DEVICE\_IP\_ADDR\_CONFIG = 1041,  
516  
517 OB\_STRUCT\_CURRENT\_DEPTH\_ALG\_MODE = 1043,  
518  
519 OB\_STRUCT\_DEPTH\_PRECISION\_SUPPORT\_LIST = 1045,  
520  
521 OB\_STRUCT\_DEVICE\_STATIC\_IP\_CONFIG\_RECORD = 1053,  
522  
523 OB\_STRUCT\_DEPTH\_HDR\_CONFIG = 1059,  
524  
525 OB\_STRUCT\_COLOR\_AE\_ROI = 1060,  
526  
527 OB\_STRUCT\_DEPTH\_AE\_ROI = 1061,  
528  
529 OB\_STRUCT ASIC SERIAL NUMBER = 1063,  
530  
531 OB\_STRUCT\_DISP\_OFFSET\_CONFIG = 1064,  
532  
533 OB\_STRUCT\_PRESET\_RESOLUTION\_CONFIG = 1069,  
534  
535 OB\_PROP\_COLOR\_AUTO\_EXPOSURE\_BOOL = 2000,  
536  
537 OB\_PROP\_COLOR\_EXPOSURE\_INT = 2001.

```
651     OB_PROP_COLOR_GAIN_INT = 2002,  
656     OB_PROP_COLOR_AUTO_WHITE_BALANCE_BOOL = 2003,  
661     OB_PROP_COLOR_WHITE_BALANCE_INT = 2004,  
666     OB_PROP_COLOR_BRIGHTNESS_INT = 2005,  
671     OB_PROP_COLOR_SHARPNESS_INT = 2006,  
676     OB_PROP_COLOR_SHUTTER_INT = 2007,  
681     OB_PROP_COLOR_SATURATION_INT = 2008,  
686     OB_PROP_COLOR_CONTRAST_INT = 2009,  
691     OB_PROP_COLOR_GAMMA_INT = 2010,  
696     OB_PROP_COLOR_ROLL_INT = 2011,  
701     OB_PROP_COLOR_AUTO_EXPOSURE_PRIORITY_INT = 2012,  
706     OB_PROP_COLOR_BACKLIGHT_COMPENSATION_INT = 2013,  
711     OB_PROP_COLOR_HUE_INT = 2014,  
716     OB_PROP_COLOR_POWER_LINE_FREQUENCY_INT = 2015,  
721     OB_PROP_DEPTH_AUTO_EXPOSURE_BOOL = 2016,  
726     OB_PROP_DEPTH_EXPOSURE_INT = 2017,  
731     OB_PROP_DEPTH_GAIN_INT = 2018,  
736     OB_PROP_IR_AUTO_EXPOSURE_BOOL = 2025,  
741     OB_PROP_IR_EXPOSURE_INT = 2026,  
746     OB_PROP_IR_GAIN_INT = 2027,  
751     OB_PROP_IR_CHANNEL_DATA_SOURCE_INT = 2028,  
756     OB_PROP_DEPTH_RM_FILTER_BOOL = 2029,  
761     OB_PROP_COLOR_MAXIMAL_GAIN_INT = 2030,  
766     OB_PROP_COLOR_MAXIMAL_SHUTTER_INT = 2031,  
771     OB_PROP_IR_SHORT_EXPOSURE_BOOL = 2032,  
776     OB_PROP_COLOR_HDR_BOOL = 2034,  
781     OB_PROP_IR_LONG_EXPOSURE_BOOL = 2035,  
786     OB_PROP_SKIP_FRAME_BOOL = 2036,  
791     OB_PROP_HDR_MERGE_BOOL = 2037,  
796     OB_PROP_COLOR_FOCUS_INT = 2038,  
800     OB_PROP_IR_RECTIFY_BOOL = 2040,  
805     OB_PROP_DEPTH_AUTO_EXPOSURE_PRIORITY_INT = 2052,  
811     OB_PROP_SDK_DISPARITY_TO_DEPTH_BOOL = 3004,
```

```

816
820     OB_PROP_SDK_DEPTH_FRAME_UNPACK_BOOL = 3007,
821
825     OB_PROP_SDK_IR_FRAME_UNPACK_BOOL = 3008,
826
830     OB_PROP_SDK_ACCEL_FRAME_TRANSFORMED_BOOL = 3009,
831
835     OB_PROP_SDK_GYRO_FRAME_TRANSFORMED_BOOL = 3010,
836
840     OB_PROP_SDK_IR_LEFT_FRAME_UNPACK_BOOL = 3011,
841
845     OB_PROP_SDK_IR_RIGHT_FRAME_UNPACK_BOOL = 3012,
846
850     OB_PROP_NETWORK_BANDWIDTH_TYPE_INT = 3027,
851
855     OB_PROP_DEVICE_PERFORMANCE_MODE_INT = 3028,
856
860     OB_RAW_DATA_CAMERA_CALIB_JSON_FILE = 4029,
861
865     OB_PROP_DEBUG_ESGM_CONFIDENCE_FLOAT = 5013,
866 } OBPropertyID,
867     ob_property_id;
868
869 // For backward compatibility
870 #define OB_PROP_TIMER_RESET_TRIGGLE_OUT_ENABLE_BOOL OB_PROP_TIMER_RESET_TRIGGER_OUT_ENABLE_BOOL
871 #define OB_PROP_LASER_ON_OFF_MODE_INT OB_PROP_LASER_ON_OFF_PATTERN_INT
872 #define OB_PROP_LASER_ENERGY_LEVEL_INT OB_PROP_LASER_POWER_LEVEL_CONTROL_INT
873 #define OB_PROP_LASER_HW_ENERGY_LEVEL_INT OB_PROP_LASER_POWER_ACTUAL_LEVEL_INT
874 #define OB_PROP_DEVICE_USB3_REPEAT_IDENTITY_BOOL OB_PROP_DEVICE_USB2_REPEAT_IDE_NOTIFY_BOOL
875 #define OB_PROP_DEPTH_SOFT_FILTER_BOOL OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_BOOL
876 #define OB_PROP_DEPTH_MAX_DIFF_INT OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_DIFF_INT
877 #define OB_PROP_DEPTH_MAX_SPECKLE_SIZE_INT OB_PROP_DEPTH_NOISE_REMOVAL_FILTER_MAX_SPECKLE_SIZE_INT
878
882 typedef enum OB.PropertyType {
883     OB_BOOL_PROPERTY = 0,
884     OB_INT_PROPERTY = 1,
885     OB_FLOAT_PROPERTY = 2,
886     OB_STRUCT_PROPERTY = 3,
887 } OB.PropertyType,
888     ob_property_type;
889
893 typedef struct OB.PropertyItem {
894     OBPropertyID id;
895     const char *name;
896     OB.PropertyType type;
897     OBPermissionType permission;
898 } OB.PropertyItem, ob_property_item;
899
900 #ifdef __cplusplus
901 }
902 #endif

```

# RecordPlayback.h File Reference

```
#include "Device.h"
```

Go to the source code of this file.

## Functions

```
OB_EXPORT ob_record_device * ob_create_record_device (ob_device *device, const char  
*file_path, bool compression_enabled, ob_error **error)  
Create a recording device for the specified device with a specified  
file path and compression enabled.
```

```
OB_EXPORT void ob_delete_record_device (ob_record_device *recorder, ob_error  
**error)  
Delete a recording device.
```

```
OB_EXPORT void ob_record_device_pause (ob_record_device *recorder, ob_error  
**error)  
Pause recording on the specified recording device.
```

```
OB_EXPORT void ob_record_device_resume (ob_record_device *recorder, ob_error  
**error)  
Resume recording on the specified recording device.
```

```
OB_EXPORT ob_device * ob_create_playback_device (const char *file_path, ob_error  
**error)  
Create a playback device for the specified file path.
```

```
OB_EXPORT void ob_playback_device_pause (ob_device *player, ob_error **error)  
Pause playback on the specified playback device.
```

```
OB_EXPORT void ob_playback_device_resume (ob_device *player, ob_error  
**error)  
Resume playback on the specified playback device.
```

```
OB_EXPORT void ob_playback_device_seek (ob_device *player, const uint64_t  
timestamp, ob_error **error)  
Set the playback to a specified time point of the played data.
```

```
OB_EXPORT void ob_playback_device_set_playback_rate (ob_device *player, const  
float rate, ob_error **error)  
Set the playback to a specified time point of the played data.
```

```
OB_EXPORT ob_playback_status ob_playback_device_get_current_playback_status (ob_device  
*player, ob_error **error)  
Get the current playback status of the played data.
```

```

OB_EXPORT void ob_playback_device_set_playback_status_changed_callback
            (ob_device *player, ob_playback_status_changed_callback
             callback, void *user_data, ob_error **error)
    Set a callback function to receive playback status updates.

OB_EXPORT uint64_t ob_playback_device_get_position (ob_device *player, ob_error
                                         **error)
    Get the current playback position of the played data.

OB_EXPORT uint64_t ob_playback_device_get_duration (ob_device *player, ob_error
                                         **error)
    Get the duration of the played data.

```

## Function Documentation

### ◆ **ob\_create\_record\_device()**

```

OB_EXPORT ob_record_device * ob_create_record_device ( ob_device * device,
                                                 const char * file_path,
                                                 bool      compression_enabled,
                                                 ob_error ** error )

```

Create a recording device for the specified device with a specified file path and compression enabled.

#### Parameters

[in] <b>device</b>	The device to record.
[in] <b>file_path</b>	The file path to record to.
[in] <b>compression_enabled</b>	Whether to enable compression for the recording.
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

#### Returns

A pointer to the newly created recording device, or NULL if an error occurred.

Referenced by **ob::RecordDevice::RecordDevice()**.

### ◆ **ob\_delete\_record\_device()**

```
OB_EXPORT void ob_delete_record_device ( ob_record_device * recorder,  
                                         ob_error **          error )
```

Delete a recording device.

#### Parameters

- [in] **recorder** The recording device to delete.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::RecordDevice::~RecordDevice\(\)](#).

#### ◆ [ob\\_record\\_device\\_pause\(\)](#)

```
OB_EXPORT void ob_record_device_pause ( ob_record_device * recorder,  
                                         ob_error **          error )
```

Pause recording on the specified recording device.

#### Parameters

- [in] **recorder** The recording device to pause.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::RecordDevice::pause\(\)](#).

#### ◆ [ob\\_record\\_device\\_resume\(\)](#)

```
OB_EXPORT void ob_record_device_resume ( ob_record_device * recorder,  
                                         ob_error **          error )
```

Resume recording on the specified recording device.

#### Parameters

- [in] **recorder** The recording device to resume.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::RecordDevice::resume\(\)](#).

#### ◆ [ob\\_create\\_playback\\_device\(\)](#)

```
OB_EXPORT ob_device * ob_create_playback_device ( const char * file_path,  
                                              ob_error ** error )
```

Create a playback device for the specified file path.

### Parameters

- [in] **file\_path** The file path to playback from.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

A pointer to the newly created playback device, or NULL if an error occurred.

Referenced by [ob::PlaybackDevice::PlaybackDevice\(\)](#).

### ◆ [ob\\_playback\\_device\\_pause\(\)](#)

```
OB_EXPORT void ob_playback_device_pause ( ob_device * player,  
                                         ob_error ** error )
```

Pause playback on the specified playback device.

### Parameters

- [in] **player** The playback device to pause.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::PlaybackDevice::pause\(\)](#).

### ◆ [ob\\_playback\\_device\\_resume\(\)](#)

```
OB_EXPORT void ob_playback_device_resume ( ob_device * player,  
                                         ob_error ** error )
```

Resume playback on the specified playback device.

### Parameters

- [in] **player** The playback device to resume.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::PlaybackDevice::resume\(\)](#).

◆ [ob\\_playback\\_device\\_seek\(\)](#)

```
OB_EXPORT void ob_playback_device_seek ( ob_device * player,  
                                         const uint64_t timestamp,  
                                         ob_error ** error )
```

Set the playback to a specified time point of the played data.

**Parameters**

- [in] **player** The playback device to set the position for.
- [in] **timestamp** The position to set the playback to, in milliseconds.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::PlaybackDevice::seek\(\)](#).

◆ [ob\\_playback\\_device\\_set\\_playback\\_rate\(\)](#)

```
OB_EXPORT void ob_playback_device_set_playback_rate ( ob_device * player,  
                                                 const float rate,  
                                                 ob_error ** error )
```

Set the playback to a specified time point of the played data.

**Parameters**

- [in] **player** The playback device to set the position for.
- [in] **rate** The playback rate to set.
- [out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::PlaybackDevice::setPlaybackRate\(\)](#).

◆ [ob\\_playback\\_device\\_get\\_current\\_playback\\_status\(\)](#)

Get the current playback status of the played data.

## Parameters

[in] **player** The playback device to get the status for.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

The current playback status of the played data.

Referenced by [ob::PlaybackDevice::getPlaybackStatus\(\)](#).

- #### ◆ ob\_playback\_device\_set\_playback\_status\_changed\_callback()

**OB\_EXPORT void**

Set a callback function to receive playback status updates.

## Parameters

[in] **player** The playback device to set the callback for.

[in] **callback** The callback function to receive playback status updates.

**[out] error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::PlaybackDevice::setPlaybackStatusChangeCallback\(\)](#).

- #### ◆ ob\_playback\_device\_get\_position()

```
OB_EXPORT uint64_t ob_playback_device_get_position ( ob_device * player,  
                                                 ob_error ** error )
```

Get the current playback position of the played data.

#### Parameters

- [in] **player** The playback device to get the position for.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The current playback position of the played data, in milliseconds.

Referenced by [ob::PlaybackDevice::getPosition\(\)](#).

### ◆ [ob\\_playback\\_device\\_get\\_duration\(\)](#)

```
OB_EXPORT uint64_t ob_playback_device_get_duration ( ob_device * player,  
                                                 ob_error ** error )
```

Get the duration of the played data.

#### Parameters

- [in] **player** The playback device to get the duration for.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The duration of the played data, in milliseconds.

Referenced by [ob::PlaybackDevice::getDuration\(\)](#).

# RecordPlayback.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4
5
6
7
8
9
10 #pragma once
11
12 #ifndef __cplusplus
13 extern "C" {
14 #endif
15
16 #include "Device.h"
17
18
19
20
21
22
23
24
25
26
27 OB_EXPORT ob_record_device *ob_create_record_device(ob_device *device, const char *file_path, bool compression_enabled, ob_error **error);
28
29
30
31
32
33
34
35 OB_EXPORT void ob_delete_record_device(ob_record_device *recorder, ob_error **error);
36
37
38
39
40
41
42
43 OB_EXPORT void ob_record_device_pause(ob_record_device *recorder, ob_error **error);
44
45
46
47
48
49
50
51 OB_EXPORT void ob_record_device_resume(ob_record_device *recorder, ob_error **error);
52
53
54
55
56
57
58
59
60 OB_EXPORT ob_device *ob_create_playback_device(const char *file_path, ob_error **error);
61
62
63
64
65
66
67
68 OB_EXPORT void ob_playback_device_pause(ob_device *player, ob_error **error);
69
70
71
72
73
74
75
76 OB_EXPORT void ob_playback_device_resume(ob_device *player, ob_error **error);
77
78
79
80
81
82
83
84
85 OB_EXPORT void ob_playback_device_seek(ob_device *player, const uint64_t timestamp, ob_error **error);
86
87
88
89
90
91
92
93
94 OB_EXPORT void ob_playback_device_set_playback_rate(ob_device *player, const float rate, ob_error **error);
95
96
97
98
99
100
101
102
103 OB_EXPORT ob_playback_status ob_playback_device_get_current_playback_status(ob_device *player, ob_error **error);
104
105
106
107
108
109
110
111
112 OB_EXPORT void ob_playback_device_set_playback_status_changed_callback(ob_device *player, ob_playback_status_changed_callback callback, void *user_data, ob_error **error);
113
114
115
116
117
118
119
120
121
122 OB_EXPORT uint64_t ob_playback_device_get_position(ob_device *player, ob_error **error);
123
124
125
126
127
128
129
130
131 OB_EXPORT uint64_t ob_playback_device_get_duration(ob_device *player, ob_error **error);
132
133
134
135 #endif
```

# RecordPlayback.hpp File Reference

Record and playback device-related types, including interfaces to create recording and playback devices, record and playback streaming data, etc. [More...](#)

```
#include "Types.hpp"
#include "Error.hpp"
#include "libobsensor/h/RecordPlayback.h"
#include "libobsensor/hpp/Device.hpp"
```

[Go to the source code of this file.](#)

## Classes

```
class ob::RecordDevice
class ob::PlaybackDevice
```

## Namespaces

```
namespace ob
```

## Typedefs

```
typedef std::function< void(OBPlaybackStatus status)> ob::PlaybackStatusChangeCallback
```

## Detailed Description

Record and playback device-related types, including interfaces to create recording and playback devices, record and playback streaming data, etc.

Definition in file [RecordPlayback.hpp](#).

## RecordPlayback.hpp

[Go to the documentation of this file.](#)

```

1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4
5 #pragma once
6
7
8 #include "Types.hpp"
9 #include "Error.hpp"
10 #include "libobsensor/h/RecordPlayback.h"
11 #include "libobsensor.hpp/Device.hpp"
12
13 namespace ob {
14
15     typedef std::function<void(OBPlaybackStatus status)> PlaybackStatusChangeCallback;
16
17     class RecordDevice {
18         private:
19             ob_record_device_t *impl_ = nullptr;
20
21         public:
22             explicit RecordDevice(std::shared_ptr<Device> device, const std::string &file, bool compressionEnabled
23                                 = true) {
24                 ob_error *error = nullptr;
25                 impl_ = ob_create_record_device(device->getImpl(), file.c_str(), compressionEnabled, &error);
26                 Error::handle(&error);
27             }
28
29             virtual ~RecordDevice() noexcept {
30                 ob_error *error = nullptr;
31                 ob_delete_record_device(impl_, &error);
32                 Error::handle(&error, false);
33             }
34
35             RecordDevice(RecordDevice &&other) noexcept {
36                 if(this != &other) {
37                     impl_ = other.impl_;
38                     other.impl_ = nullptr;
39                 }
40             }
41
42             RecordDevice &operator=(RecordDevice &&other) noexcept {
43                 if(this != &other) {
44                     impl_ = other.impl_;
45                     other.impl_ = nullptr;
46                 }
47
48                 return *this;
49             }
50
51             RecordDevice(const RecordDevice &) = delete;
52             RecordDevice &operator=(const RecordDevice &) = delete;
53
54             public:
55                 void pause() {
56                     ob_error *error = nullptr;
57                     ob_record_device_pause(impl_, &error);
58                     Error::handle(&error);
59                 }
60
61             }
62

```

```

63
64 void resume() {
65     ob_error *error = nullptr;
66     ob_record_device_resume(impl_, &error);
67     Error::handle(&error);
68 }
69 };
70
71 class PlaybackDevice : public Device {
72 public:
73     explicit PlaybackDevice(const std::string &file) : Device(nullptr) {
74         ob_error *error = nullptr;
75         impl_ = ob_create_playback_device(file.c_str(), &error);
76         Error::handle(&error);
77     }
78
79     virtual ~PlaybackDevice() noexcept override = default;
80
81     PlaybackDevice(PlaybackDevice &&other) noexcept : Device(std::move(other)) {}
82
83     PlaybackDevice &operator=(PlaybackDevice &&other) noexcept {
84         Device::operator=(std::move(other));
85         return *this;
86     }
87
88     PlaybackDevice(const PlaybackDevice &) = delete;
89     PlaybackDevice &operator=(const PlaybackDevice &) = delete;
90
91 public:
92     void pause() {
93         ob_error *error = nullptr;
94         ob_playback_device_pause(impl_, &error);
95         Error::handle(&error);
96     }
97
98     void resume() {
99         ob_error *error = nullptr;
100        ob_playback_device_resume(impl_, &error);
101        Error::handle(&error);
102    }
103
104    void seek(const int64_t timestamp) {
105        ob_error *error = nullptr;
106        ob_playback_device_seek(impl_, timestamp, &error);
107        Error::handle(&error);
108    }
109
110    void setPlaybackRate(const float rate) {
111        ob_error *error = nullptr;
112        ob_playback_device_set_playback_rate(impl_, rate, &error);
113        Error::handle(&error);
114    }
115
116    void setPlaybackStatusChangeCallback(PlaybackStatusChangeCallback callback) {
117        callback_ = callback;
118        ob_error *error = nullptr;
119        ob_playback_device_set_playback_status_changed_callback(impl_, &PlaybackDevice::playbackStatusCallback, this, &error);
120        Error::handle(&error);
121    }
122
123    OBPlaybackStatus getPlaybackStatus() const {
124        ob_error *error = nullptr;
125        OBPlaybackStatus status = ob_playback_device_get_current_playback_status(impl_, &error);
126        Error::handle(&error);
127    }

```

```

149     return status;
150 }
151
152 uint64_t getPosition() const {
153     ob_error *error = nullptr;
154     uint64_t position = ob_playback_device_get_position(impl_, &error);
155     Error::handle(&error);
156
157     return position;
158 }
159
160 uint64_t getDuration() const {
161     ob_error *error = nullptr;
162     uint64_t duration = ob_playback_device_get_duration(impl_, &error);
163     Error::handle(&error);
164
165     return duration;
166 }
167
168 private:
169     static void playbackStatusCallback(OBPlaybackStatus status, void *userData) {
170         auto *playbackDevice = static_cast<PlaybackDevice *>(userData);
171         if(playbackDevice && playbackDevice->callback_) {
172             playbackDevice->callback_(status);
173         }
174     }
175
176     PlaybackStatusChangeCallback callback_;
177 };
178 } // namespace ob

```

# Sensor.h File Reference

Defines types related to sensors, used for obtaining stream configurations, opening and closing streams, and setting and getting sensor properties. [More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_sensor_list_get_sensor_count ob_sensor_list_get_count  
#define ob_sensor_get_recommended_filter_list ob_sensor_create_recommended_filter_list
```

## Functions

```
OB_EXPORT ob_sensor_type ob_sensor_get_type (const ob_sensor *sensor, ob_error  
**error)
```

Get the type of the sensor.

```
OB_EXPORT ob_stream_profile_list * ob_sensor_get_stream_profile_list (const ob_sensor  
*sensor, ob_error **error)
```

Get a list of all supported stream profiles.

```
OB_EXPORT void ob_sensor_start (ob_sensor *sensor, const  
ob_stream_profile *profile, ob_frame_callback callback, void  
*user_data, ob_error **error)
```

Open the current sensor and set the callback data frame.

```
OB_EXPORT void ob_sensor_stop (ob_sensor *sensor, ob_error **error)
```

Stop the sensor stream.

```
OB_EXPORT void ob_sensor_switch_profile (ob_sensor *sensor,  
ob_stream_profile *profile, ob_error **error)
```

Switch resolutions.

```
OB_EXPORT void ob_delete_sensor (ob_sensor *sensor, ob_error **error)
```

Delete a sensor object.

```
OB_EXPORT ob_filter_list * ob_sensor_create_recommended_filter_list (const  
ob_sensor *sensor, ob_error **error)
```

Create a list of recommended filters for the specified sensor.

```
OB_EXPORT uint32_t ob_sensor_list_get_count (const ob_sensor_list *sensor_list,  
ob_error **error)
```

Get the number of sensors in the sensor list.

```
OB_EXPORT ob_sensor_type ob_sensor_list_get_sensor_type (const ob_sensor_list
    *sensor_list, uint32_t index, ob_error **error)
    Get the sensor type.
```

```
OB_EXPORT ob_sensor * ob_sensor_list_get_sensor_by_type (const ob_sensor_list
    *sensor_list, ob_sensor_type sensorType, ob_error **error)
    Get a sensor by sensor type.
```

```
OB_EXPORT ob_sensor * ob_sensor_list_get_sensor (const ob_sensor_list
    *sensor_list, uint32_t index, ob_error **error)
    Get a sensor by index number.
```

```
OB_EXPORT void ob_delete_sensor_list (ob_sensor_list *sensor_list, ob_error
    **error)
    Delete a list of sensor objects.
```

## Detailed Description

Defines types related to sensors, used for obtaining stream configurations, opening and closing streams, and setting and getting sensor properties.

Definition in file **Sensor.h**.

## Macro Definition Documentation

### ◆ **ob\_sensor\_list\_get\_sensor\_count**

```
#define ob_sensor_list_get_sensor_count ob_sensor_list_get_count
```

Definition at line **127** of file **Sensor.h**.

### ◆ **ob\_sensor\_get\_recommended\_filter\_list**

```
#define ob_sensor_get_recommended_filter_list ob_sensor_create_recommended_filter_list
```

Definition at line **128** of file **Sensor.h**.

## Function Documentation

◆ **ob\_sensor\_get\_type()**

```
OB_EXPORT ob_sensor_type ob_sensor_get_type ( const ob_sensor * sensor,  
                                              ob_error **      error )
```

Get the type of the sensor.

**Parameters**

- [in] **sensor** The sensor object.
- [out] **error** Logs error messages.

**Returns**

The sensor type.

Referenced by [ob::Sensor::getType\(\)](#).

◆ **ob\_sensor\_get\_stream\_profile\_list()**

```
OB_EXPORT ob_stream_profile_list * ob_sensor_get_stream_profile_list ( const ob_sensor * sensor,  
                                                               ob_error **      error )
```

Get a list of all supported stream profiles.

**Parameters**

- [in] **sensor** The sensor object.
- [out] **error** Logs error messages.

**Returns**

A list of stream profiles.

Referenced by [ob::Sensor::getStreamProfileList\(\)](#).

◆ **ob\_sensor\_start()**

```
OB_EXPORT void ob_sensor_start ( ob_sensor * sensor,  
                                const ob_stream_profile * profile,  
                                ob_frame_callback callback,  
                                void * user_data,  
                                ob_error ** error )
```

Open the current sensor and set the callback data frame.

### Parameters

- [in] **sensor** The sensor object.
- [in] **profile** The stream configuration information.
- [in] **callback** The callback function triggered when frame data arrives.
- [in] **user\_data** Any user data to pass in and get from the callback.
- [out] **error** Logs error messages.

Referenced by [ob::Sensor::start\(\)](#).

### ◆ [ob\\_sensor\\_stop\(\)](#)

```
OB_EXPORT void ob_sensor_stop ( ob_sensor * sensor,  
                                ob_error ** error )
```

Stop the sensor stream.

### Parameters

- [in] **sensor** The sensor object.
- [out] **error** Logs error messages.

Referenced by [ob::Sensor::stop\(\)](#).

### ◆ [ob\\_sensor\\_switch\\_profile\(\)](#)

```
OB_EXPORT void ob_sensor_switch_profile ( ob_sensor * sensor,  
                                         ob_stream_profile * profile,  
                                         ob_error ** error )
```

Switch resolutions.

#### Parameters

- [in] **sensor** The sensor object.
- [in] **profile** The stream configuration information.
- [out] **error** Logs error messages.

Referenced by [ob::Sensor::switchProfile\(\)](#).

#### ◆ [ob\\_delete\\_sensor\(\)](#)

```
OB_EXPORT void ob_delete_sensor ( ob_sensor * sensor,  
                                    ob_error ** error )
```

Delete a sensor object.

#### Parameters

- [in] **sensor** The sensor object to delete.
- [out] **error** Logs error messages.

Referenced by [ob::Sensor::operator=\(\)](#), and [ob::Sensor::~Sensor\(\)](#).

#### ◆ [ob\\_sensor\\_create\\_recommended\\_filter\\_list\(\)](#)

Create a list of recommended filters for the specified sensor.

## Parameters

[in] **sensor** The **ob\_sensor** object.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

## ob filter list

Referenced by [ob::Sensor::createRecommendedFilters\(\)](#).

- #### ◆ ob\_sensor\_list\_get\_count()

Get the number of sensors in the sensor list.

## Parameters

[in] **sensor\_list** The list of sensor objects.

[out] **error** Logs error messages.

## Returns

The number of sensors in the list.

Referenced by **ob::SensorList::getCount()**.

- #### ◆ ob\_sensor\_list\_get\_sensor\_type()

Get the sensor type.

## Parameters

- [in] **sensor\_list** The list of sensor objects.
- [in] **index** The index of the sensor on the list.
- [out] **error** Logs error messages.

## Returns

## The sensor type.

Referenced by [ob::SensorList::getSensorType\(\)](#).

- #### ◆ ob\_sensor\_list\_get\_sensor\_by\_type()

Get a sensor by sensor type.

## Parameters

- [in] **sensor\_list** The list of sensor objects.
- [in] **sensorType** The sensor type to be obtained.
- [out] **error** Logs error messages.

## Returns

The sensor pointer. If the specified type of sensor does not exist, it will return null.

Referenced by [ob::SensorList::getSensor\(\)](#).

- ◆ ob sensor list get sensor()

```
OB_EXPORT ob_sensor* ob_sensor_list_get_sensor ( const ob_sensor_list* sensor_list,  
                                          uint32_t                          index,  
                                          ob_error **                      error )
```

Get a sensor by index number.

#### Parameters

- [in] **sensor\_list** The list of sensor objects.
- [in] **index** The index of the sensor on the list.
- [out] **error** Logs error messages.

#### Returns

The sensor object.

Referenced by [ob::SensorList::getSensor\(\)](#).

### ◆ [ob\\_delete\\_sensor\\_list\(\)](#)

```
OB_EXPORT void ob_delete_sensor_list ( ob_sensor_list* sensor_list,  
                                          ob_error **                      error )
```

Delete a list of sensor objects.

#### Parameters

- [in] **sensor\_list** The list of sensor objects to delete.
- [out] **error** Logs error messages.

Referenced by [ob::SensorList::~SensorList\(\)](#).

# Sensor.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #ifndef __cplusplus
11 extern "C" {
12 #endif
13
14 #include "ObTypes.h"
15
23 OB_EXPORT ob_sensor_type ob_sensor_get_type(const ob_sensor *sensor, ob_error **error);
24
32 OB_EXPORT ob_stream_profile_list *ob_sensor_get_stream_profile_list(const ob_sensor *sensor, ob_error **error);
33
43 OB_EXPORT void ob_sensor_start(ob_sensor *sensor, const ob_stream_profile *profile, ob_frame_callback callback, void *user_data, ob_error **error);
44
51 OB_EXPORT void ob_sensor_stop(ob_sensor *sensor, ob_error **error);
52
60 OB_EXPORT void ob_sensor_switch_profile(ob_sensor *sensor, ob_stream_profile *profile, ob_error **error);
61
68 OB_EXPORT void ob_delete_sensor(ob_sensor *sensor, ob_error **error);
69
78 OB_EXPORT ob_filter_list *ob_sensor_create_recommended_filter_list(const ob_sensor *sensor, ob_error **error);
79
87 OB_EXPORT uint32_t ob_sensor_list_get_count(const ob_sensor_list *sensor_list, ob_error **error);
88
97 OB_EXPORT ob_sensor_type ob_sensor_list_get_sensor_type(const ob_sensor_list *sensor_list, uint32_t index, ob_error **error);
98
107 OB_EXPORT ob_sensor *ob_sensor_list_get_sensor_by_type(const ob_sensor_list *sensor_list, ob_sensor_type sensorType, ob_error **error);
108
117 OB_EXPORT ob_sensor *ob_sensor_list_get_sensor(const ob_sensor_list *sensor_list, uint32_t index, ob_error **error);
118
125 OB_EXPORT void ob_delete_sensor_list(ob_sensor_list *sensor_list, ob_error **error);
126
127 #define ob_sensor_list_get_sensor_count ob_sensor_list_get_count
128 #define ob_sensor_get_recommended_filter_list ob_sensor_create_recommended_filter_list
129
130 #ifdef __cplusplus
131 }
132 #endif
```

# Sensor.hpp File Reference

Defines types related to sensors, which are used to obtain stream configurations, open and close streams, and set and get sensor properties. [More...](#)

```
#include "Types.hpp"
#include "libobsensor.hpp/Filter.hpp"
#include "libobsensor/h/Sensor.h"
#include "libobsensor/h/Filter.h"
#include "Error.hpp"
#include "StreamProfile.hpp"
#include "Device.hpp"
#include "Frame.hpp"
#include <functional>
#include <memory>
#include <vector>
```

[Go to the source code of this file.](#)

## Classes

```
class ob::Sensor
class ob::SensorList
```

## Namespaces

```
namespace ob
```

## Detailed Description

Defines types related to sensors, which are used to obtain stream configurations, open and close streams, and set and get sensor properties.

Definition in file [Sensor.hpp](#).

# Sensor.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #include "Types.hpp"
11 #include "libobsensor.hpp/Filter.hpp"
12 #include "libobsensor/h/Sensor.h"
13 #include "libobsensor/h/Filter.h"
14 #include "Error.hpp"
15 #include "StreamProfile.hpp"
16 #include "Device.hpp"
17 #include "Frame.hpp"
18 #include <functional>
19 #include <memory>
20 #include <vector>
21
22 namespace ob {
23
24 class Sensor {
25 public:
31     typedef std::function<void(std::shared_ptr<Frame> frame)> FrameCallback;
32
33 protected:
34     ob_sensor_t *impl_;
35     FrameCallback callback_;
36
37 public:
38     explicit Sensor(ob_sensor_t *impl) : impl_(impl) {}
39
40     Sensor(Sensor &&sensor) noexcept : impl_(sensor.impl_) {
41         sensor.impl_ = nullptr;
42     }
43
44     Sensor &operator=(Sensor &&sensor) noexcept {
45         if(this != &sensor) {
46             ob_error *error = nullptr;
47             ob_delete_sensor(impl_, &error);
48             Error::handle(&error);
49             impl_ = sensor.impl_;
50             sensor.impl_ = nullptr;
51         }
52         return *this;
53     }
54
55     Sensor(const Sensor &sensor)      = delete;
56     Sensor &operator=(const Sensor &sensor) = delete;
57
58     virtual ~Sensor() noexcept {
59         ob_error *error = nullptr;
60         ob_delete_sensor(impl_, &error);
61         Error::handle(&error, false);
62     }
63
69     OBSensorType getType() const {
70         ob_error *error = nullptr;
71         auto type = ob_sensor_get_type(impl_, &error);
72         Error::handle(&error);
```

```

73     return type;
74 }
75
81 std::shared_ptr<StreamProfileList> getStreamProfileList() const {
82     ob_error *error = nullptr;
83     auto list = ob_sensor_get_stream_profile_list(impl_, &error);
84     Error::handle(&error);
85     return std::make_shared<StreamProfileList>(list);
86 }
87
88 std::vector<std::shared_ptr<Filter>> createRecommendedFilters() const {
89     ob_error *error = nullptr;
90     auto list = ob_sensor_create_recommended_filter_list(impl_, &error);
91     Error::handle(&error);
92     auto filter_count = ob_filter_list_get_count(list, &error);
93
94     std::vector<std::shared_ptr<Filter>> filters;
95     for(uint32_t i = 0; i < filter_count; i++) {
96         auto filterImpl = ob_filter_list_get_filter(list, i, &error);
97         Error::handle(&error);
98         filters.push_back(std::make_shared<Filter>(filterImpl));
99     }
100    ob_delete_filter_list(list, &error);
101    Error::handle(&error, false);
102    return filters;
103 }
104
105 void start(std::shared_ptr<StreamProfile> streamProfile, FrameCallback callback) {
106     ob_error *error = nullptr;
107     callback_ = std::move(callback);
108     ob_sensor_start(impl_, const_cast<ob_stream_profile_t *>(streamProfile->getImpl()), &Sensor::frame
109     Callback, this, &error);
110     Error::handle(&error);
111 }
112
113 void stop() const {
114     ob_error *error = nullptr;
115     ob_sensor_stop(impl_, &error);
116     Error::handle(&error);
117 }
118
119 void switchProfile(std::shared_ptr<StreamProfile> streamProfile) {
120     ob_error *error = nullptr;
121     ob_sensor_switch_profile(impl_, const_cast<ob_stream_profile_t *>(streamProfile->getImpl()), &erro
122     r);
123     Error::handle(&error);
124 }
125
126 private:
127     static void frameCallback(ob_frame *frame, void *userData) {
128         auto sensor = static_cast<Sensor*>(userData);
129         sensor->callback_(std::make_shared<Frame>(frame));
130     }
131
132 public:
133     // The following interfaces are deprecated and are retained here for compatibility purposes.
134     OBSensorType type() const {
135         return getType();
136     }
137
138     std::vector<std::shared_ptr<Filter>> getRecommendedFilters() const {
139         return createRecommendedFilters();
140     }
141
142 };
143

```

```

160 class SensorList {
161 private:
162     ob_sensor_list_t *impl_ = nullptr;
163
164 public:
165     explicit SensorList(ob_sensor_list_t *impl) : impl_(impl) {}
166
167     ~SensorList() noexcept {
168         ob_error *error = nullptr;
169         ob_delete_sensor_list(impl_, &error);
170         Error::handle(&error, false);
171     }
172
173     uint32_t getCount() const {
174         ob_error *error = nullptr;
175         auto count = ob_sensor_list_get_count(impl_, &error);
176         Error::handle(&error);
177         return count;
178     }
179
180     OBsensorType getSensorType(uint32_t index) const {
181         ob_error *error = nullptr;
182         auto type = ob_sensor_list_get_sensor_type(impl_, index, &error);
183         Error::handle(&error);
184         return type;
185     }
186
187     std::shared_ptr<Sensor> getSensor(uint32_t index) const {
188         ob_error *error = nullptr;
189         auto sensor = ob_sensor_list_get_sensor(impl_, index, &error);
190         Error::handle(&error);
191         return std::make_shared<Sensor>(sensor);
192     }
193
194     std::shared_ptr<Sensor> getSensor(OBsensorType sensorType) const {
195         ob_error *error = nullptr;
196         auto sensor = ob_sensor_list_get_sensor_by_type(impl_, sensorType, &error);
197         Error::handle(&error);
198         return std::make_shared<Sensor>(sensor);
199     }
200
201 public:
202     // The following interfaces are deprecated and are retained here for compatibility purposes.
203     uint32_t count() const {
204         return getCount();
205     }
206
207     OBsensorType type(uint32_t index) const {
208         return getSensorType(index);
209     }
210 };
211
212 } // namespace ob
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236

```

# StreamProfile.h File Reference

The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream. [More...](#)

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Macros

```
#define ob_stream_profile_format ob_stream_profile_get_format  
#define ob_stream_profile_type ob_stream_profile_get_type  
#define ob_video_stream_profile_fps ob_video_stream_profile_get_fps  
#define ob_video_stream_profile_width ob_video_stream_profile_get_width  
#define ob_video_stream_profile_height ob_video_stream_profile_get_height  
#define ob_accel_stream_profile_full_scale_range ob_accel_stream_profile_get_full_scale_range  
#define ob_accel_stream_profile_sample_rate ob_accel_stream_profile_get_sample_rate  
#define ob_gyro_stream_profile_full_scale_range ob_gyro_stream_profile_get_full_scale_range  
#define ob_gyro_stream_profile_sample_rate ob_gyro_stream_profile_get_sample_rate  
#define ob_stream_profile_list_count ob_stream_profile_list_get_count
```

## Functions

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_create\_stream\_profile** (**ob\_stream\_type** type,  
**ob\_format** format, **ob\_error** \*\*error)

Create a stream profile object.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_create\_video\_stream\_profile** (**ob\_stream\_type** type,  
**ob\_format** format, uint32\_t width, uint32\_t height, uint32\_t  
fps, **ob\_error** \*\*error)

Create a video stream profile object.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_create\_accel\_stream\_profile**  
(**ob\_accel\_full\_scale\_range** full\_scale\_range,  
**ob\_accel\_sample\_rate** sample\_rate, **ob\_error** \*\*error)

Create a accel stream profile object.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_create\_gyro\_stream\_profile** (**ob\_gyro\_full\_scale\_range**  
full\_scale\_range, **ob\_gyro\_sample\_rate** sample\_rate,  
**ob\_error** \*\*error)

Create a gyro stream profile object.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_create\_stream\_profile\_from\_other\_stream\_profile** (const **ob\_stream\_profile** \*srcProfile, **ob\_error** \*\*error)  
Copy the stream profile object from an other stream profile object.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_create\_stream\_profile\_with\_new\_format** (const **ob\_stream\_profile** \*profile, **ob\_format** new\_format, **ob\_error** \*\*error)  
Copy the stream profile object with a new format object.

**OB\_EXPORT** void **ob\_delete\_stream\_profile** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Delete the stream configuration.

**OB\_EXPORT** **ob\_format** **ob\_stream\_profile\_get\_format** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Get stream profile format.

**OB\_EXPORT** void **ob\_stream\_profile\_set\_format** (**ob\_stream\_profile** \*profile, **ob\_format** format, **ob\_error** \*\*error)  
Set stream profile format.

**OB\_EXPORT** **ob\_stream\_type** **ob\_stream\_profile\_get\_type** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Get stream profile type.

**OB\_EXPORT** void **ob\_stream\_profile\_set\_type** (const **ob\_stream\_profile** \*profile, **ob\_stream\_type** type, **ob\_error** \*\*error)  
Set stream profile type.

**OB\_EXPORT** **ob\_extrinsic** **ob\_stream\_profile\_get\_extrinsic\_to** (const **ob\_stream\_profile** \*source, **ob\_stream\_profile** \*target, **ob\_error** \*\*error)  
Get the extrinsic for source stream to target stream.

**OB\_EXPORT** void **ob\_stream\_profile\_set\_extrinsic\_to** (**ob\_stream\_profile** \*source, const **ob\_stream\_profile** \*target, **ob\_extrinsic** extrinsic, **ob\_error** \*\*error)  
Set the extrinsic for source stream to target stream.

**OB\_EXPORT** void **ob\_stream\_profile\_set\_extrinsic\_to\_type** (**ob\_stream\_profile** \*source, const **ob\_stream\_type** type, **ob\_extrinsic** extrinsic, **ob\_error** \*\*error)  
Set the extrinsic for source stream to target stream type.

**OB\_EXPORT** uint32\_t **ob\_video\_stream\_profile\_get\_fps** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Get the frame rate of the video stream.

**OB\_EXPORT** uint32\_t **ob\_video\_stream\_profile\_get\_width** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the width of the video stream.

**OB\_EXPORT** void **ob\_video\_stream\_profile\_set\_width** (**ob\_stream\_profile**\*profile, uint32\_t width, **ob\_error** \*\*error)

Set the width of the video stream.

**OB\_EXPORT** uint32\_t **ob\_video\_stream\_profile\_get\_height** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the height of the video stream.

**OB\_EXPORT** void **ob\_video\_stream\_profile\_set\_height** (**ob\_stream\_profile**\*profile, uint32\_t height, **ob\_error** \*\*error)

Set the height of the video stream.

**OB\_EXPORT** **ob\_camera\_intrinsic** **ob\_video\_stream\_profile\_get\_intrinsic** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the intrinsic of the video stream profile.

**OB\_EXPORT** void **ob\_video\_stream\_profile\_set\_intrinsic** (**ob\_stream\_profile**\*profile, **ob\_camera\_intrinsic** intrinsic, **ob\_error** \*\*error)

Set the intrinsic of the video stream profile.

**OB\_EXPORT** **ob\_camera\_distortion** **ob\_video\_stream\_profile\_get\_distortion** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the distortion of the video stream profile.

**OB\_EXPORT** void **ob\_video\_stream\_profile\_set\_distortion** (**ob\_stream\_profile** \*profile, **ob\_camera\_distortion** distortion, **ob\_error** \*\*error)

Set the distortion of the video stream profile.

**OB\_EXPORT** **ob\_disparity\_param** **ob\_disparity\_based\_stream\_profile\_get\_disparity\_param** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the process param of the disparity stream.

**OB\_EXPORT** void **ob\_disparity\_based\_stream\_profile\_set\_disparity\_param** (**ob\_stream\_profile** \*profile, **ob\_disparity\_param** param, **ob\_error** \*\*error)

Set the disparity process param of the disparity stream.

**OB\_EXPORT** **ob\_accel\_full\_scale\_range** **ob\_accel\_stream\_profile\_get\_full\_scale\_range** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the full-scale range of the accelerometer stream.

**OB\_EXPORT** **ob\_accel\_sample\_rate** **ob\_accel\_stream\_profile\_get\_sample\_rate** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the sampling frequency of the accelerometer frame.

**OB\_EXPORT** **ob\_accel\_intrinsic** **ob\_accel\_stream\_profile\_get\_intrinsic** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)

Get the intrinsic of the accelerometer stream.

**OB\_EXPORT** void **ob\_accel\_stream\_profile\_set\_intrinsic** (**ob\_stream\_profile**\*profile, **ob\_accel\_intrinsic** intrinsic, **ob\_error** \*\*error)  
Set the intrinsic of the accelerometer stream.

**OB\_EXPORT** **ob\_gyro\_full\_scale\_range** **ob\_gyro\_stream\_profile\_get\_full\_scale\_range** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Get the full-scale range of the gyroscope stream.

**OB\_EXPORT** **ob\_gyro\_sample\_rate** **ob\_gyro\_stream\_profile\_get\_sample\_rate** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Get the sampling frequency of the gyroscope stream.

**OB\_EXPORT** **ob\_gyro\_intrinsic** **ob\_gyro\_stream\_get\_intrinsic** (const **ob\_stream\_profile** \*profile, **ob\_error** \*\*error)  
Get the intrinsic of the gyroscope stream.

**OB\_EXPORT** void **ob\_gyro\_stream\_set\_intrinsic** (**ob\_stream\_profile** \*profile, **ob\_gyro\_intrinsic** intrinsic, **ob\_error** \*\*error)  
Set the intrinsic of the gyroscope stream.

**OB\_EXPORT** uint32\_t **ob\_stream\_profile\_list\_get\_count** (const **ob\_stream\_profile\_list** \*profile\_list, **ob\_error** \*\*error)  
Get the number of StreamProfile lists.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_stream\_profile\_list\_get\_profile** (const **ob\_stream\_profile\_list** \*profile\_list, int index, **ob\_error** \*\*error)  
Get the corresponding StreamProfile by subscripting.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_stream\_profile\_list\_get\_video\_stream\_profile** (const **ob\_stream\_profile\_list** \*profile\_list, int width, int height, **ob\_format** format, int fps, **ob\_error** \*\*error)  
Match the corresponding **ob\_stream\_profile** through the passed parameters. If there are multiple matches, the first one in the list will be returned by default. If no matched profile is found, an error will be returned.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_stream\_profile\_list\_get\_accel\_stream\_profile** (const **ob\_stream\_profile\_list** \*profile\_list, **ob\_accel\_full\_scale\_range** full\_scale\_range, **ob\_accel\_sample\_rate** sample\_rate, **ob\_error** \*\*error)  
Match the corresponding **ob\_stream\_profile** through the passed parameters. If there are multiple matches, the first one in the list will be returned by default. If no matched profile is found, an error will be returned.

**OB\_EXPORT** **ob\_stream\_profile** \* **ob\_stream\_profile\_list\_get\_gyro\_stream\_profile** (const **ob\_stream\_profile\_list** \*profile\_list, **ob\_gyro\_full\_scale\_range** full\_scale\_range, **ob\_gyro\_sample\_rate** sample\_rate, **ob\_error** \*\*error)

Match the corresponding **ob\_stream\_profile** through the passed parameters. If there are multiple matches, the first one in the list will be returned by default. If no matched profile is found, an error will be returned.

**OB\_EXPORT** void **ob\_delete\_stream\_profile\_list** (const **ob\_stream\_profile\_list** \*profile\_list, **ob\_error** \*\*error)

Delete the stream profile list.

## Detailed Description

The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream.

Definition in file **StreamProfile.h**.

## Macro Definition Documentation

### ◆ **ob\_stream\_profile\_format**

```
#define ob_stream_profile_format ob_stream_profile_get_format
```

Definition at line **404** of file **StreamProfile.h**.

### ◆ **ob\_stream\_profile\_type**

```
#define ob_stream_profile_type ob_stream_profile_get_type
```

Definition at line **405** of file **StreamProfile.h**.

### ◆ **ob\_video\_stream\_profile\_fps**

```
#define ob_video_stream_profile_fps ob_video_stream_profile_get_fps
```

Definition at line **406** of file **StreamProfile.h**.

### ◆ **ob\_video\_stream\_profile\_width**

```
#define ob_video_stream_profile_width ob_video_stream_profile_get_width
```

Definition at line **407** of file **StreamProfile.h**.

◆ **ob\_video\_stream\_profile\_height**

```
#define ob_video_stream_profile_height ob_video_stream_profile_get_height
```

Definition at line **408** of file **StreamProfile.h**.

◆ **ob\_accel\_stream\_profile\_full\_scale\_range**

```
#define ob_accel_stream_profile_full_scale_range ob_accel_stream_profile_get_full_scale_range
```

Definition at line **409** of file **StreamProfile.h**.

◆ **ob\_accel\_stream\_profile\_sample\_rate**

```
#define ob_accel_stream_profile_sample_rate ob_accel_stream_profile_get_sample_rate
```

Definition at line **410** of file **StreamProfile.h**.

◆ **ob\_gyro\_stream\_profile\_full\_scale\_range**

```
#define ob_gyro_stream_profile_full_scale_range ob_gyro_stream_profile_get_full_scale_range
```

Definition at line **411** of file **StreamProfile.h**.

◆ **ob\_gyro\_stream\_profile\_sample\_rate**

```
#define ob_gyro_stream_profile_sample_rate ob_gyro_stream_profile_get_sample_rate
```

Definition at line **412** of file **StreamProfile.h**.

◆ **ob\_stream\_profile\_list\_count**

```
#define ob_stream_profile_list_count ob_stream_profile_list_get_count
```

Definition at line **413** of file **StreamProfile.h**.

## Function Documentation

### ◆ **ob\_create\_stream\_profile()**

```
OB_EXPORT ob_stream_profile * ob_create_stream_profile ( ob_stream_type type,  
                                 ob_format       format,  
                                 ob_error **  error )
```

Create a stream profile object.

#### Parameters

[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_stream\_profile\*** return the stream profile object

### ◆ **ob\_create\_video\_stream\_profile()**

Create a video stream profile object.

### Parameters

- [in] **type** Stream type
- [in] **format** Stream format
- [in] **width** Stream width
- [in] **height** Stream height
- [in] **fps** Stream frame rate
- [out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

ob stream profile\* return the video stream profile object

#### ◆ ob create accel stream profile()

Create a accel stream profile object.

## Parameters

[in] **full\_scale\_range** Accel full scale range  
[in] **sample\_rate** Accel sample rate  
[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

ob stream profile\* return the accel stream profile object

#### ◆ ob create gyro stream profile()

Create a gyro stream profile object.

## Parameters

[in] **full\_scale\_range** Gyro full scale range

[in] **sample\_rate** Gyro sample rate

**[out] error** Pointer to an error object that will be set if an error occurs.

## Returns

`ob_stream_profile*` return the accel stream profile object

- #### ◆ ob\_create\_stream\_profile\_from\_other\_stream\_profile()

**OB\_EXPORT ob\_stream\_profile \***

Copy the stream profile object from an other stream profile object.

## Parameters

[in] **srcProfile** Source stream profile object

**[out] error** Pointer to an error object that will be set if an error occurs.

## Returns

`ob_stream_profile*` return the new stream profile object

- #### ◆ ob\_create\_stream\_profile\_with\_new\_format()

```
OB_EXPORT ob_stream_profile *
ob_create_stream_profile_with_new_format
( const ob_stream_profile * profile,
  ob_format           new_format,
  ob_error **        error )
```

Copy the stream profile object with a new format object.

#### Parameters

[in] **profile** Stream profile object  
[in] **new\_format** New format  
[out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

`ob_stream_profile*` return the new stream profile object with the new format

### ◆ `ob_delete_stream_profile()`

```
OB_EXPORT void ob_delete_stream_profile ( const ob_stream_profile * profile,
                                         ob_error **           error )
```

Delete the stream configuration.

#### Parameters

[in] **profile** Stream profile object .  
[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::StreamProfile::operator=()**, and **ob::StreamProfile::~StreamProfile()**.

### ◆ `ob_stream_profile_get_format()`

```
OB_EXPORT ob_format ob_stream_profile_get_format ( const ob_stream_profile * profile,  
                                              ob_error ** error )
```

Get stream profile format.

#### Parameters

- [in] **profile** Stream profile object
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

**ob\_format** return the format of the stream

Referenced by **ob::StreamProfile::getFormat()**.

#### ◆ ob\_stream\_profile\_set\_format()

```
OB_EXPORT void ob_stream_profile_set_format ( ob_stream_profile * profile,  
                                              ob_format format,  
                                              ob_error ** error )
```

Set stream profile format.

#### Parameters

- [in] **profile** Stream profile object
- [in] **format** The format of the stream
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### ◆ ob\_stream\_profile\_get\_type()

Get stream profile type.

## Parameters

[in] **profile** Stream profile object

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_stream\_type** stream type

Referenced by `ob::StreamProfileFactory::create()`, and `ob::StreamProfile::getType()`.

- #### ◆ ob\_stream\_profile\_set\_type()

Set stream profile type.

## Parameters

[in] **profile** Stream profile object

[in] **type** The type of the stream

[out] **error** Pointer to an error object that will be set if an error occurs.

- #### ◆ ob\_stream\_profile\_get\_extrinsic\_to()

Get the extrinsic for source stream to target stream.

## Parameters

[in] **source** Source stream profile

[in] **target** Target stream profile

**[out] error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_extrinsic** The extrinsic

Referenced by [ob::StreamProfile::getExtrinsicTo\(\)](#).

#### ◆ ob\_stream\_profile\_set\_extrinsic\_to()

Set the extrinsic for source stream to target stream.

## Parameters

[in] **source** Stream profile object

[in] **target** Target stream type

[in] **extrinsic** The extrinsic

**[out] error** Pointer to an error object that will be set if an error occurs.

Referenced by [ob::StreamProfile::bindExtrinsicTo\(\)](#).

#### ◆ ob\_stream\_profile\_set\_extrinsic\_to\_type0

```
OB_EXPORT void ob_stream_profile_set_extrinsic_to_type ( ob_stream_profile * source,  
                                         const ob_stream_type type,  
                                         ob_extrinsic          extrinsic,  
                                         ob_error **           error )
```

Set the extrinsic for source stream to target stream type.

### Parameters

- [in] **source** Source stream profile
- [in] **type** Target stream type
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_extrinsic** The extrinsic

Referenced by **ob::StreamProfile::bindExtrinsicTo()**.

◆ [ob\\_video\\_stream\\_profile\\_get\\_fps\(\)](#)

```
OB_EXPORT uint32_t ob_video_stream_profile_get_fps ( const ob_stream_profile * profile,  
                                                 ob_error **                  error )
```

Get the frame rate of the video stream.

### Parameters

- [in] **profile** Stream profile object
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

uint32\_t return the frame rate of the stream

Referenced by **ob::VideoStreamProfile::getFps()**.

◆ [ob\\_video\\_stream\\_profile\\_get\\_width\(\)](#)

Get the width of the video stream.

## Parameters

- [in] **profile** Stream profile object , If the profile is not a video stream configuration, an error will be returned
- [out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

uint32\_t return the width of the stream

Referenced by [ob::VideoStreamProfile::getWidth\(\)](#).

- #### ◆ ob video stream profile set width()

Set the width of the video stream.

## Parameters

- [in] **profile** Stream profile object , If the profile is not a video stream configuration, an error will be returned
- [in] **width** The width of the stream
- [out] **error** Pointer to an error object that will be set if an error occurs.

- #### ◆ ob\_video\_stream\_profile\_get\_height()

Get the height of the video stream.

## Parameters

- [in] **profile** Stream profile object , If the profile is not a video stream configuration, an error will be returned
- [out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

`uint32_t` return the height of the stream

Referenced by [ob::VideoStreamProfile::getHeight\(\)](#).

- #### ◆ ob\_video\_stream\_profile\_set\_height()

Set the height of the video stream.

## Parameters

- [in] **profile** Stream profile object , If the profile is not a video stream configuration, an error will be returned
- [in] **height** The height of the stream
- [out] **error** Pointer to an error object that will be set if an error occurs.

- #### ◆ ob video stream profile get intrinsic()

Get the intrinsic of the video stream profile.

## Parameters

[in] **profile** Stream profile object

**[out] error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob camera intrinsic** Return the intrinsic of the stream

Referenced by **ob::VideoStreamProfile::getIntrinsic()**.

- #### ◆ ob\_video\_stream\_profile\_set\_intrinsic()

Set the intrinsic of the video stream profile.

## Parameters

[in] **profile** Stream profile object

[in] **intrinsic** The intrinsic of the stream

**[out] error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::VideoStreamProfile::setIntrinsic()**.

- #### ◆ ob\_video\_stream\_profile\_get\_distortion()

## **OB\_EXPORT ob\_camera\_distortion**

```
ob_video_stream_profile_get_distortion ( const ob_stream_profile * profile,
                                         ob_error ** error )
```

Get the distortion of the video stream profile.

### Parameters

[in] **profile** Stream profile object

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_camera\_distortion** Return the distortion of the stream

Referenced by **ob::VideoStreamProfile::getDistortion()**.

## ◆ **ob\_video\_stream\_profile\_set\_distortion()**

```
OB_EXPORT void ob_video_stream_profile_set_distortion ( ob_stream_profile * profile,
                                                       ob_camera_distortion distortion,
                                                       ob_error ** error )
```

Set the distortion of the video stream profile.

### Parameters

[in] **profile** Stream profile object

[in] **distortion** The distortion of the stream

[out] **error** Pointer to an error object that will be set if an error occurs.

Referenced by **ob::VideoStreamProfile::setDistortion()**.

## ◆ **ob\_disparity\_based\_stream\_profile\_get\_disparity\_param()**

## **OB\_EXPORT ob\_disparity\_param**

```
ob_disparity_based_stream_profile_get_disparity_param( const ob_stream_profile * profile,  
                                                    ob_error ** error )
```

Get the process param of the disparity stream.

### Parameters

- [in] **profile** Stream profile object
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

**ob\_disparity\_param** Return the disparity process param of the stream

## ◆ **ob\_disparity\_based\_stream\_profile\_set\_disparity\_param()**

```
OB_EXPORT void ob_disparity_based_stream_profile_set_disparity_param( ob_stream_profile * profile,  
                                                               ob_disparity_param param,  
                                                               ob_error ** error )
```

Set the disparity process param of the disparity stream.

### Parameters

- [in] **profile** Stream profile object. If the profile is not for the disparity stream, an error will be returned.
- [in] **param** The disparity process param of the disparity stream.
- [out] **error** Pointer to an error object that will be set if an error occurs.

## ◆ **ob\_accel\_stream\_profile\_get\_full\_scale\_range()**

**OB\_EXPORT ob\_accel\_full\_scale\_range**  
ob\_accel\_stream\_profile\_get\_full\_scale\_range  
( const **ob\_stream\_profile** \* profile,  
**ob\_error** \*\* error )

Get the full-scale range of the accelerometer stream.

#### Parameters

- [in] **profile** Stream profile object. If the profile is not for the accelerometer stream, an error will be returned.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The full-scale range of the accelerometer stream.

Referenced by [ob::AccelStreamProfile::getFullScaleRange\(\)](#).

◆ [ob\\_accel\\_stream\\_profile\\_get\\_sample\\_rate\(\)](#)

**OB\_EXPORT ob\_accel\_sample\_rate**  
ob\_accel\_stream\_profile\_get\_sample\_rate  
( const **ob\_stream\_profile** \* profile,  
**ob\_error** \*\* error )

Get the sampling frequency of the accelerometer frame.

#### Parameters

- [in] **profile** Stream profile object. If the profile is not for the accelerometer stream, an error will be returned.
- [out] **error** Pointer to an error object that will be set if an error occurs.

#### Returns

The sampling frequency of the accelerometer frame.

Referenced by [ob::AccelStreamProfile::getSampleRate\(\)](#).

◆ [ob\\_accel\\_stream\\_profile\\_get\\_intrinsic\(\)](#)

Get the intrinsic of the accelerometer stream.

## Parameters

[in] **profile** Stream profile object. If the profile is not for the accelerometer stream, an error will be returned.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_accel\_intrinsic** Return the intrinsic of the accelerometer stream.

Referenced by [ob::AccelStreamProfile::getIntrinsic\(\)](#).

#### ◆ ob accel stream profile set intrinsic()

Set the intrinsic of the accelerometer stream.

## Parameters

[in] **profile** Stream profile object. If the profile is not for the accelerometer stream, an error will be returned.

[in] **intrinsic** The intrinsic of the accelerometer stream.

**[out] error** Pointer to an error object that will be set if an error occurs.

#### ◆ ob\_gyro\_stream\_profile\_get\_full\_scale\_range()

**OB\_EXPORT ob\_gyro\_full\_scale\_range**  
ob\_gyro\_stream\_profile\_get\_full\_scale\_range  
( const **ob\_stream\_profile** \* profile,  
**ob\_error** \*\* error )

Get the full-scale range of the gyroscope stream.

### Parameters

- [in] **profile** Stream profile object. If the profile is not for the gyroscope stream, an error will be returned.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The full-scale range of the gyroscope stream.

Referenced by [ob::GyroStreamProfile::getFullScaleRange\(\)](#).

◆ [ob\\_gyro\\_stream\\_profile\\_get\\_sample\\_rate\(\)](#)

**OB\_EXPORT ob\_gyro\_sample\_rate**  
ob\_gyro\_stream\_profile\_get\_sample\_rate  
( const **ob\_stream\_profile** \* profile,  
**ob\_error** \*\* error )

Get the sampling frequency of the gyroscope stream.

### Parameters

- [in] **profile** Stream profile object. If the profile is not for the gyroscope stream, an error will be returned.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The sampling frequency of the gyroscope stream.

Referenced by [ob::GyroStreamProfile::getSampleRate\(\)](#).

◆ [ob\\_gyro\\_stream\\_get\\_intrinsic\(\)](#)

Get the intrinsic of the gyroscope stream.

## Parameters

[in] **profile** Stream profile object. If the profile is not for the gyroscope stream, an error will be returned.

[out] **error** Pointer to an error object that will be set if an error occurs.

## Returns

**ob\_gyro\_intrinsic** Return the intrinsic of the gyroscope stream.

Referenced by [ob::GyroStreamProfile::getIntrinsic\(\)](#).

- #### ◆ ob\_gyro\_stream set\_intrinsic()

Set the intrinsic of the gyroscope stream.

## Parameters

- [in] **profile** Stream profile object. If the profile is not for the gyroscope stream, an error will be returned.
- [in] **intrinsic** The intrinsic of the gyroscope stream.
- [out] **error** Pointer to an error object that will be set if an error occurs.

- ◆ ob stream profile list get count()

```
OB_EXPORT uint32_t ob_stream_profile_list_get_count ( const ob_stream_profile_list * profile_list,  
                                              ob_error **  
                                              error )
```

Get the number of StreamProfile lists.

### Parameters

[in] **profile\_list** StreamProfile list.

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The number of StreamProfile lists.

Referenced by **ob::StreamProfileList::getCount()**.

### ◆ **ob\_stream\_profile\_list\_get\_profile()**

```
OB_EXPORT ob_stream_profile *  
ob_stream_profile_list_get_profile  
                                ( const ob_stream_profile_list * profile_list,  
                                int  
                                index,  
                                ob_error **  
                                error )
```

Get the corresponding StreamProfile by subscripting.

### Attention

The stream profile returned by this function should be deleted by calling **ob\_delete\_stream\_profile()** when it is no longer needed.

### Parameters

[in] **profile\_list** StreamProfile lists.

[in] **index** Index.

[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The matching profile.

Referenced by **ob::StreamProfileList::getProfile()**.

### ◆ **ob\_stream\_profile\_list\_get\_video\_stream\_profile()**

```

OB_EXPORT ob_stream_profile *
ob_stream_profile_list_get_video_stream_profile
( const ob_stream_profile_list* profile_list,
  int width,
  int height,
  ob_format format,
  int fps,
  ob_error ** error )

```

Match the corresponding **ob\_stream\_profile** through the passed parameters. If there are multiple matches, the first one in the list will be returned by default. If no matched profile is found, an error will be returned.

### Attention

The stream profile returned by this function should be deleted by calling **ob\_delete\_stream\_profile()** when it is no longer needed.

### Parameters

- [in] **profile\_list** Resolution list.
- [in] **width** Width. If you don't need to add matching conditions, you can pass OB\_WIDTH\_ANY.
- [in] **height** Height. If you don't need to add matching conditions, you can pass OB\_HEIGHT\_ANY.
- [in] **format** Format. If you don't need to add matching conditions, you can pass OB\_FORMAT\_ANY.
- [in] **fps** Frame rate. If you don't need to add matching conditions, you can pass OB\_FPS\_ANY.
- [out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The matching profile.

Referenced by **ob::StreamProfileList::getVideoStreamProfile()**.

- ◆ **ob\_stream\_profile\_list\_get\_accel\_stream\_profile()**

```
OB_EXPORT ob_stream_profile *
ob_stream_profile_list_get_accel_stream_profile
( const ob_stream_profile_list* profile_list,
  ob_accel_full_scale_range full_scale_range,
  ob_accel_sample_rate sample_rate,
  ob_error ** error )
```

Match the corresponding **ob\_stream\_profile** through the passed parameters. If there are multiple matches, the first one in the list will be returned by default. If no matched profile is found, an error will be returned.

### Attention

The stream profile returned by this function should be deleted by calling **ob\_delete\_stream\_profile()** when it is no longer needed.

### Parameters

[in] **profile\_list** Resolution list.  
[in] **full\_scale\_range** Full-scale range. If you don't need to add matching conditions, you can pass 0.  
[in] **sample\_rate** Sample rate. If you don't need to add matching conditions, you can pass 0.  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

The matching profile.

Referenced by **ob::StreamProfileList::getAccelStreamProfile()**.

- ◆ **ob\_stream\_profile\_list\_get\_gyro\_stream\_profile()**

```

OB_EXPORT ob_stream_profile *
ob_stream_profile_list_get_gyro_stream_profile
( const ob_stream_profile_list* profile_list,
  ob_gyro_full_scale_range      full_scale_range,
  ob_gyro_sample_rate         sample_rate,
  ob_error **                error )

```

Match the corresponding **ob\_stream\_profile** through the passed parameters. If there are multiple matches, the first one in the list will be returned by default. If no matched profile is found, an error will be returned.

### Attention

The stream profile returned by this function should be deleted by calling **ob\_delete\_stream\_profile()** when it is no longer needed.

### Parameters

[in] <b>profile_list</b>	Resolution list.
[in] <b>full_scale_range</b>	Full-scale range. If you don't need to add matching conditions, you can pass 0.
[in] <b>sample_rate</b>	Sample rate. If you don't need to add matching conditions, you can pass 0.
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

The matching profile.

Referenced by **ob::StreamProfileList::getGyroStreamProfile()**.

### ◆ **ob\_delete\_stream\_profile\_list()**

```

OB_EXPORT void ob_delete_stream_profile_list ( const ob_stream_profile_list* profile_list,
                                              ob_error **                  error )

```

Delete the stream profile list.

### Parameters

[in] <b>profile_list</b>	Stream configuration list.
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

Referenced by **ob::StreamProfileList::~StreamProfileList()**.

# StreamProfile.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6 #ifndef __cplusplus
7 extern "C" {
8 #endif
9
10 #include "ObTypes.h"
11
12 OB_EXPORT ob_stream_profile *ob_create_stream_profile(ob_stream_type type, ob_format format, ob_error **error);
13
14 OB_EXPORT ob_stream_profile *ob_create_video_stream_profile(ob_stream_type type, ob_format format,
15                 uint32_t width, uint32_t height, uint32_t fps,
16                 ob_error **error);
17
18 OB_EXPORT ob_stream_profile *ob_create_accel_stream_profile(ob_accel_full_scale_range full_scale_range,
19                 ob_accel_sample_rate sample_rate, ob_error **error);
20
21 OB_EXPORT ob_stream_profile *ob_create_gyro_stream_profile(ob_gyro_full_scale_range full_scale_range,
22                 ob_gyro_sample_rate sample_rate, ob_error **error);
23
24 OB_EXPORT ob_stream_profile *ob_create_stream_profile_from_other_stream_profile(const ob_stream_profile *srcProfile, ob_error **error);
25
26 OB_EXPORT ob_stream_profile *ob_create_stream_profile_with_new_format(const ob_stream_profile *profile, ob_format new_format, ob_error **error);
27
28 OB_EXPORT void ob_delete_stream_profile(const ob_stream_profile *profile, ob_error **error);
29
30 OB_EXPORT ob_format ob_stream_profile_get_format(const ob_stream_profile *profile, ob_error **error);
31 ;
32
33 OB_EXPORT void ob_stream_profile_set_format(ob_stream_profile *profile, ob_format format, ob_error **error);
34
35 OB_EXPORT ob_stream_type ob_stream_profile_get_type(const ob_stream_profile *profile, ob_error **error);
36
37 OB_EXPORT void ob_stream_profile_set_type(const ob_stream_profile *profile, ob_stream_type type, ob_error **error);
38
39 OB_EXPORT ob_extrinsic ob_stream_profile_get_extrinsic_to(const ob_stream_profile *source, ob_stream_profile *target, ob_error **error);
40
41 OB_EXPORT void ob_stream_profile_set_extrinsic_to(ob_stream_profile *source, const ob_stream_profile *target, ob_extrinsic extrinsic, ob_error **error);
42
43 OB_EXPORT void ob_stream_profile_set_extrinsic_to_type(ob_stream_profile *source, const ob_stream_type type, ob_extrinsic extrinsic, ob_error **error);
44
45 OB_EXPORT uint32_t ob_video_stream_profile_get_fps(const ob_stream_profile *profile, ob_error **error);
46
47 OB_EXPORT uint32_t ob_video_stream_profile_get_width(const ob_stream_profile *profile, ob_error **error);
```

```

171
179 OB_EXPORT void ob_video_stream_profile_set_width(ob_stream_profile *profile, uint32_t width, ob_error **error);
180
188 OB_EXPORT uint32_t ob_video_stream_profile_get_height(const ob_stream_profile *profile, ob_error **error);
189
197 OB_EXPORT void ob_video_stream_profile_set_height(ob_stream_profile *profile, uint32_t height, ob_error **error);
198
206 OB_EXPORT ob_camera_intrinsic ob_video_stream_profile_get_intrinsic(const ob_stream_profile *profile, ob_error **error);
207
215 OB_EXPORT void ob_video_stream_profile_set_intrinsic(ob_stream_profile *profile, ob_camera_intrinsic *intrinsic, ob_error **error);
216
224 OB_EXPORT ob_camera_distortion ob_video_stream_profile_get_distortion(const ob_stream_profile *profile, ob_error **error);
225
233 OB_EXPORT void ob_video_stream_profile_set_distortion(ob_stream_profile *profile, ob_camera_distortion *distortion, ob_error **error);
234
242 OB_EXPORT ob_disparity_param ob_disparity_based_stream_profile_get_disparity_param(const ob_stream_profile *profile, ob_disparity_param *param, ob_error **error);
243
251 OB_EXPORT void ob_disparity_based_stream_profile_set_disparity_param(ob_stream_profile *profile, ob_disparity_param *param, ob_error **error);
252
260 OB_EXPORT ob_accel_full_scale_range ob_accel_stream_profile_get_full_scale_range(const ob_stream_profile *profile, ob_error **error);
261
269 OB_EXPORT ob_accel_sample_rate ob_accel_stream_profile_get_sample_rate(const ob_stream_profile *profile, ob_error **error);
270
278 OB_EXPORT ob_accel_intrinsic ob_accel_stream_profile_get_intrinsic(const ob_stream_profile *profile, ob_error **error);
279
287 OB_EXPORT void ob_accel_stream_profile_set_intrinsic(ob_stream_profile *profile, ob_accel_intrinsic *intrinsic, ob_error **error);
288
296 OB_EXPORT ob_gyro_full_scale_range ob_gyro_stream_profile_get_full_scale_range(const ob_stream_profile *profile, ob_error **error);
297
305 OB_EXPORT ob_gyro_sample_rate ob_gyro_stream_profile_get_sample_rate(const ob_stream_profile *profile, ob_error **error);
306
314 OB_EXPORT ob_gyro_intrinsic ob_gyro_stream_get_intrinsic(const ob_stream_profile *profile, ob_error **error);
315
323 OB_EXPORT void ob_gyro_stream_set_intrinsic(ob_stream_profile *profile, ob_gyro_intrinsic *intrinsic, ob_error **error);
324
332 OB_EXPORT uint32_t ob_stream_profile_list_get_count(const ob_stream_profile_list *profile_list, ob_error **error);
333
344 OB_EXPORT ob_stream_profile *ob_stream_profile_list_get_profile(const ob_stream_profile_list *profile_list, int index, ob_error **error);
345
360 OB_EXPORT ob_stream_profile *ob_stream_profile_list_get_video_stream_profile(const ob_stream_profile_list *profile_list, int width, int height,
361                                     ob_format format, int fps, ob_error **error);
362
375 OB_EXPORT ob_stream_profile *ob_stream_profile_list_get_accel_stream_profile(const ob_stream_profile_list *profile_list,
376                                     ob_accel_full_scale_range full_scale_range, ob_accel_sample

```

```

377     _rate sample_rate,
378                               ob_error **error);
391 OB_EXPORT ob_stream_profile *ob_stream_profile_list_get_gyro_stream_profile(const ob_stream_profi
392     le_list *profile_list,
393                               ob_gyro_full_scale_range full_scale_range, ob_gyro_sample_r
394     ate sample_rate,
395                               ob_error **error);
401 OB_EXPORT void ob_delete_stream_profile_list(const ob_stream_profile_list *profile_list, ob_error **err
402     or);
403 // The following interfaces are deprecated and are retained here for compatibility purposes.
404 #define ob_stream_profile_format ob_stream_profile_get_format
405 #define ob_stream_profile_type ob_stream_profile_get_type
406 #define ob_video_stream_profile_fps ob_video_stream_profile_get_fps
407 #define ob_video_stream_profile_width ob_video_stream_profile_get_width
408 #define ob_video_stream_profile_height ob_video_stream_profile_get_height
409 #define ob_accel_stream_profile_full_scale_range ob_accel_stream_profile_get_full_scale_range
410 #define ob_accel_stream_profile_sample_rate ob_accel_stream_profile_get_sample_rate
411 #define ob_gyro_stream_profile_full_scale_range ob_gyro_stream_profile_get_full_scale_range
412 #define ob_gyro_stream_profile_sample_rate ob_gyro_stream_profile_get_sample_rate
413 #define ob_stream_profile_list_count ob_stream_profile_list_get_count
414
415 #ifdef __cplusplus
416 }
417 #endif

```

# StreamProfile.hpp File Reference

The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream. [More...](#)

```
#include "Types.hpp"
#include "libobsensor/h/StreamProfile.h"
#include "libobsensor/h/Error.h"
#include <iostream>
#include <memory>
```

[Go to the source code of this file.](#)

## Classes

class [\*\*ob::StreamProfile\*\*](#)

class [\*\*ob::VideoStreamProfile\*\*](#)

    Class representing a video stream profile. [More...](#)

class [\*\*ob::AccelStreamProfile\*\*](#)

    Class representing an accelerometer stream profile. [More...](#)

class [\*\*ob::GyroStreamProfile\*\*](#)

    Class representing a gyroscope stream profile. [More...](#)

class [\*\*ob::StreamProfileFactory\*\*](#)

class [\*\*ob::StreamProfileList\*\*](#)

## Namespaces

namespace [\*\*ob\*\*](#)

## Detailed Description

The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream.

Definition in file [\*\*StreamProfile.hpp\*\*](#).

# StreamProfile.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #include "Types.hpp"
11 #include "libobsensor/h/StreamProfile.h"
12 #include "libobsensor/h/Error.h"
13 #include <iostream>
14 #include <memory>
15
16 namespace ob {
17
18 class StreamProfile : public std::enable_shared_from_this<StreamProfile> {
19 protected:
20     const ob_stream_profile_t *impl_ = nullptr;
21
22 public:
23     StreamProfile(StreamProfile &streamProfile) = delete;
24     StreamProfile &operator=(StreamProfile &streamProfile) = delete;
25
26     StreamProfile(StreamProfile &&streamProfile) noexcept : impl_(streamProfile.impl_) {
27         streamProfile.impl_ = nullptr;
28     }
29
30     StreamProfile &operator=(StreamProfile &&streamProfile) noexcept {
31         if(this != &streamProfile) {
32             ob_error *error = nullptr;
33             ob_delete_stream_profile(impl_, &error);
34             Error::handle(&error);
35             impl_ = streamProfile.impl_;
36             streamProfile.impl_ = nullptr;
37         }
38         return *this;
39     }
40
41     virtual ~StreamProfile() noexcept {
42         if(impl_) {
43             ob_error *error = nullptr;
44             ob_delete_stream_profile(impl_, &error);
45             Error::handle(&error);
46         }
47     }
48
49     const ob_stream_profile *getImpl() const {
50         return impl_;
51     }
52
53     OBFormat getFormat() const {
54         ob_error *error = nullptr;
55         auto format = ob_stream_profile_get_format(impl_, &error);
56         Error::handle(&error);
57         return format;
58     }
59
60     OBStreamType getType() const {
61         ob_error *error = nullptr;
62         auto type = ob_stream_profile_get_type(impl_, &error);
63     }
64 }
```



```

178     auto    fps = ob_video_stream_profile_get_fps(impl_, &error);
179     Error::handle(&error);
180     return fps;
181 }
182
183 uint32_t getWidth() const {
184     ob_error *error=nullptr;
185     auto    width = ob_video_stream_profile_get_width(impl_, &error);
186     Error::handle(&error);
187     return width;
188 }
189
190 uint32_t getHeight() const {
191     ob_error *error =nullptr;
192     auto    height = ob_video_stream_profile_get_height(impl_, &error);
193     Error::handle(&error);
194     return height;
195 }
196
197 OBCameraIntrinsic getIntrinsic() const {
198     ob_error *error =nullptr;
199     auto    intrinsic = ob_video_stream_profile_get_intrinsic(impl_, &error);
200     Error::handle(&error);
201     return intrinsic;
202 }
203
204 void setIntrinsic(const OBCameraIntrinsic &intrinsic) {
205     ob_error *error =nullptr;
206     ob_video_stream_profile_set_intrinsic(const_cast<ob_stream_profile_t *>(impl_), intrinsic, &error);
207     Error::handle(&error);
208 }
209
210 OBCameraDistortion getDistortion() const {
211     ob_error *error =nullptr;
212     auto    distortion = ob_video_stream_profile_get_distortion(impl_, &error);
213     Error::handle(&error);
214     return distortion;
215 }
216
217 void setDistortion(const OBCameraDistortion &distortion) {
218     ob_error *error =nullptr;
219     ob_video_stream_profile_set_distortion(const_cast<ob_stream_profile_t *>(impl_), distortion, &erro
220 r);
221     Error::handle(&error);
222 }
223
224 public:
225     // The following interfaces are deprecated and are retained here for compatibility purposes.
226     uint32_t fps() const {
227         return getFps();
228     }
229
230     uint32_t width() const {
231         return getWidth();
232     }
233
234     uint32_t height() const {
235         return getHeight();
236     }
237 };
238
239 class AccelStreamProfile : public StreamProfile {
240 public:
241     explicit AccelStreamProfile(const ob_stream_profile_t *impl) : StreamProfile(impl) {}
242
243     ^ ~~~~~

```

```

270     ~OACcelStreamProfile() noexcept override = default;
271
272     OBAccelFullScaleRange getFullScaleRange() const {
273         ob_error *error = nullptr;
274         auto fullScaleRange = ob_accel_stream_profile_get_full_scale_range(impl_, &error);
275         Error::handle(&error);
276         return fullScaleRange;
277     }
278
279
280     OBAccelSampleRate getSampleRate() const {
281         ob_error *error = nullptr;
282         auto sampleRate = ob_accel_stream_profile_get_sample_rate(impl_, &error);
283         Error::handle(&error);
284         return sampleRate;
285     }
286
287
288     OBAccelIntrinsic getIntrinsic() const {
289         ob_error *error = nullptr;
290         auto intrinsic = ob_accel_stream_profile_get_intrinsic(impl_, &error);
291         Error::handle(&error);
292         return intrinsic;
293     }
294
295     public:
296         // The following interfaces are deprecated and are retained here for compatibility purposes.
297         OBAccelFullScaleRange fullScaleRange() const {
298             return getFullScaleRange();
299         }
300
301
302         OBAccelSampleRate sampleRate() const {
303             return getSampleRate();
304         }
305
306     };
307
308     class GyroStreamProfile : public StreamProfile {
309     public:
310         explicit GyroStreamProfile(const ob_stream_profile_t *impl) : StreamProfile(impl) {}
311
312         ~GyroStreamProfile() noexcept override = default;
313
314         OBGyroFullScaleRange getFullScaleRange() const {
315             ob_error *error = nullptr;
316             auto fullScaleRange = ob_gyro_stream_profile_get_full_scale_range(impl_, &error);
317             Error::handle(&error);
318             return fullScaleRange;
319         }
320
321         OBGyroSampleRate getSampleRate() const {
322             ob_error *error = nullptr;
323             auto sampleRate = ob_gyro_stream_profile_get_sample_rate(impl_, &error);
324             Error::handle(&error);
325             return sampleRate;
326         }
327
328         OBGyroIntrinsic getIntrinsic() const {
329             ob_error *error = nullptr;
330             auto intrinsic = ob_gyro_stream_get_intrinsic(impl_, &error);
331             Error::handle(&error);
332             return intrinsic;
333         }
334
335         public:
336             // The following interfaces are deprecated and are retained here for compatibility purposes.
337             OBGyroFullScaleRange fullScaleRange() const {
338                 return getFullScaleRange();
339             }

```

```

...
375
376     OBGyroSampleRate sampleRate() const {
377         return getSampleRate();
378     }
379 };
380
381 template <typename T> bool StreamProfile::is() const {
382     switch(this->getType()) {
383     case OB_STREAM_VIDEO:
384     case OB_STREAM_IR:
385     case OB_STREAM_IR_LEFT:
386     case OB_STREAM_IR_RIGHT:
387     case OB_STREAM_COLOR:
388     case OB_STREAM_DEPTH:
389     case OB_STREAM_RAW_PHASE:
390     case OB_STREAM_CONFIDENCE:
391         return typeid(T) == typeid(VideoStreamProfile);
392     case OB_STREAM_ACCEL:
393         return typeid(T) == typeid(AccelStreamProfile);
394     case OB_STREAM_GYRO:
395         return typeid(T) == typeid(GyroStreamProfile);
396     default:
397         break;
398     }
399     return false;
400 }
401
402 class StreamProfileFactory {
403 public:
404     static std::shared_ptr<StreamProfile> create(const ob_stream_profile_t *impl) {
405         ob_error *error = nullptr;
406         const auto type = ob_stream_profile_get_type(impl, &error);
407         Error::handle(&error);
408         switch(type) {
409         case OB_STREAM_IR:
410         case OB_STREAM_IR_LEFT:
411         case OB_STREAM_IR_RIGHT:
412         case OB_STREAM_DEPTH:
413         case OB_STREAM_COLOR:
414         case OB_STREAM_VIDEO:
415         case OB_STREAM_CONFIDENCE:
416             return std::make_shared<VideoStreamProfile>(impl);
417         case OB_STREAM_ACCEL:
418             return std::make_shared<AccelStreamProfile>(impl);
419         case OB_STREAM_GYRO:
420             return std::make_shared<GyroStreamProfile>(impl);
421         default: {
422             ob_error *err = ob_create_error(OB_STATUS_ERROR, "Unsupported stream type.", "StreamProfile"
423             Factory::create", "", OB_EXCEPTION_TYPE_INVALID_VALUE);
424             Error::handle(&err);
425             return nullptr;
426         }
427     }
428 };
429
430 class StreamProfileList {
431 protected:
432     const ob_stream_profile_list_t *impl_;
433
434 public:
435     explicit StreamProfileList(ob_stream_profile_list_t *impl) : impl_(impl) {}
436     ~StreamProfileList() noexcept {
437         ob_error *error = nullptr;
438         ob_delete_stream_profile_list(impl_, &error);

```

```

439     Error::handle(&error, false);
440 }
441
442 uint32_t getCount() const {
443     ob_error *error = nullptr;
444     auto count = ob_stream_profile_list_get_count(impl_, &error);
445     Error::handle(&error);
446     return count;
447 }
448
449 std::shared_ptr<StreamProfile> getProfile(uint32_t index) const {
450     ob_error *error = nullptr;
451     auto profile = ob_stream_profile_list_get_profile(impl_, index, &error);
452     Error::handle(&error);
453     return StreamProfileFactory::create(profile);
454 }
455
456 std::shared_ptr<VideoStreamProfile> getVideoStreamProfile(int width = OB_WIDTH_ANY, int height =
457     OB_HEIGHT_ANY, OBFormat format = OB_FORMAT_ANY,
458     int fps = OB_FPS_ANY) const {
459     ob_error *error = nullptr;
460     auto profile = ob_stream_profile_list_get_video_stream_profile(impl_, width, height, format, fps, &
461     error);
462     Error::handle(&error);
463     auto vsp = StreamProfileFactory::create(profile);
464     return vsp->as<VideoStreamProfile>();
465 }
466
467 std::shared_ptr<AccelStreamProfile> getAccelStreamProfile(OBAccelFullScaleRange fullScaleRange, OB
468     AccelSampleRate sampleRate) const {
469     ob_error *error = nullptr;
470     auto profile = ob_stream_profile_list_get_accel_stream_profile(impl_, fullScaleRange, sampleRate,
471     &error);
472     Error::handle(&error);
473     auto asp = StreamProfileFactory::create(profile);
474     return asp->as<AccelStreamProfile>();
475 }
476
477 std::shared_ptr<GyroStreamProfile> getGyroStreamProfile(OBGyroFullScaleRange fullScaleRange, OBGy
478     roSampleRate sampleRate) const {
479     ob_error *error = nullptr;
480     auto profile = ob_stream_profile_list_get_gyro_stream_profile(impl_, fullScaleRange, sampleRate,
481     &error);
482     Error::handle(&error);
483     auto gsp = StreamProfileFactory::create(profile);
484     return gsp->as<GyroStreamProfile>();
485 }
486
487 public:
488     // The following interfaces are deprecated and are retained here for compatibility purposes.
489     uint32_t count() const {
490         return getCount();
491     }
492 };
493
494 } // namespace ob

```

# TypeHelper.h File Reference

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Functions

```
OB_EXPORT const char * ob_format_type_to_string (OBFormat type)
    Convert OBFormat to "char*" type and then return.

OB_EXPORT const char * ob_frame_type_to_string (OBFrameType type)
    Convert OBFrameType to "char*" type and then return.

OB_EXPORT const char * ob_stream_type_to_string (OBStreamType type)
    Convert OBStreamType to "char*" type and then return.

OB_EXPORT const char * ob_sensor_type_to_string (OBSensorType type)
    Convert OBSensorType to "char*" type and then return.

OB_EXPORT const char * ob_imu_rate_type_to_string (OBIMUSampleRate type)
    Convert OBIMUSampleRate to "char*" type and then return.

OB_EXPORT const char * ob_gyro_range_type_to_string (OBGyroFullScaleRange type)
    Convert OBGyroFullScaleRange to "char*" type and then return.

OB_EXPORT const char * ob_accel_range_type_to_string (OBAccelFullScaleRange type)
    Convert OBAccelFullScaleRange to "char*" type and then return.

OB_EXPORT const char * ob_meta_data_type_to_string (OBFrameMetadataType type)
    Convert OBFrameMetadataType to "char*" type and then return.

OB_EXPORT OBStreamType ob_sensor_type_to_stream_type (OBSensorType type)
    Convert OBStreamType to OBSensorType.

OB_EXPORT const char * ob_format_to_string (OBFormat format)
    Convert OBFormat to "char*" type and then return.
```

## Function Documentation

- ◆ [ob\\_format\\_type\\_to\\_string\(\)](#)

**OB\_EXPORT** const char \* ob\_format\_type\_to\_string ( **OBFormat** type )

Convert **OBFormat** to "char\*" type and then return.

#### Parameters

[in] type **OBFormat** type.

#### Returns

**OBFormat** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBFormatTypeToString()**.

#### ◆ ob\_frame\_type\_to\_string()

**OB\_EXPORT** const char \* ob\_frame\_type\_to\_string ( **OBFrameType** type )

Convert **OBFrameType** to "char\*" type and then return.

#### Parameters

[in] type **OBFrameType** type.

#### Returns

**OBFrameType** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBFrameTypeToString()**.

#### ◆ ob\_stream\_type\_to\_string()

**OB\_EXPORT** const char \* ob\_stream\_type\_to\_string ( **OBStreamType** type )

Convert **OBStreamType** to "char\*" type and then return.

#### Parameters

[in] type **OBStreamType** type.

#### Returns

**OBStreamType** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBStreamTypeToString()**.

#### ◆ ob\_sensor\_type\_to\_string()

**OB\_EXPORT** const char\* ob\_sensor\_type\_to\_string ( **OBSensorType** type )

Convert **OBSensorType** to " char\* " type and then return.

#### Parameters

[in] type **OBSensorType** type.

#### Returns

**OBSensorType** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBSensorTypeToString()**.

◆ **ob\_imu\_rate\_type\_to\_string()**

**OB\_EXPORT** const char\* ob\_imu\_rate\_type\_to\_string ( **OBIMUSampleRate** type )

Convert **OBIMUSampleRate** to " char\* " type and then return.

#### Parameters

[in] type **OBIMUSampleRate** type.

#### Returns

**OBIMUSampleRate** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBIMUSampleRateTypeToString()**.

◆ **ob\_gyro\_range\_type\_to\_string()**

**OB\_EXPORT** const char\* ob\_gyro\_range\_type\_to\_string ( **OBGyroFullScaleRange** type )

Convert **OBGyroFullScaleRange** to " char\* " type and then return.

#### Parameters

[in] type **OBGyroFullScaleRange** type.

#### Returns

**OBGyroFullScaleRange** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBGyroFullScaleRangeTypeToString()**.

◆ **ob\_accel\_range\_type\_to\_string()**

**OB\_EXPORT** const char\* ob\_accel\_range\_type\_to\_string ( **OBAccelFullScaleRange** type )

Convert **OBAccelFullScaleRange** to "char\*" type and then return.

#### Parameters

[in] type **OBAccelFullScaleRange** type.

#### Returns

**OBAccelFullScaleRange** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBAccelFullScaleRangeTypeToString()**.

- ◆ **ob\_meta\_data\_type\_to\_string()**

**OB\_EXPORT** const char\* ob\_meta\_data\_type\_to\_string ( **OBFrameMetadataType** type )

Convert **OBFrameMetadataType** to "char\*" type and then return.

#### Parameters

[in] type **OBFrameMetadataType** type.

#### Returns

**OBFrameMetadataType** of "char\*" type.

Referenced by **ob::TypeHelper::convertOBFrameMetadataTypeToString()**.

- ◆ **ob\_sensor\_type\_to\_stream\_type()**

**OB\_EXPORT OBStreamType** ob\_sensor\_type\_to\_stream\_type ( **OBSensorType** type )

Convert **OBStreamType** to **OBSensorType**.

#### Parameters

[in] type The sensor type to convert.

#### Returns

**OBStreamType** The corresponding stream type.

Referenced by **ob::TypeHelper::convertSensorTypeToStreamType()**.

- ◆ **ob\_format\_to\_string()**

**OB\_EXPORT** const char\* ob\_format\_to\_string ( **OBFormat** format )

Convert **OBFormat** to "char\*" type and then return.

#### Parameters

**format** The **OBFormat** to convert.

#### Returns

The string.

# TypeHelper.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6 #ifndef __cplusplus
7 extern "C" {
8 #endif
9
10 #include "ObTypes.h"
11
18 OB_EXPORT const char* ob_format_type_to_string(OBFormat type);
19
26 OB_EXPORT const char* ob_frame_type_to_string(OBFrameType type);
27
34 OB_EXPORT const char* ob_stream_type_to_string(OBStreamType type);
35
42 OB_EXPORT const char* ob_sensor_type_to_string(OBSensorType type);
43
50 OB_EXPORT const char* ob_imu_rate_type_to_string(OBIMUSampleRate type);
51
58 OB_EXPORT const char* ob_gyro_range_type_to_string(OBGyroFullScaleRange type);
59
66 OB_EXPORT const char* ob_accel_range_type_to_string(OBAccelFullScaleRange type);
67
74 OB_EXPORT const char* ob_meta_data_type_to_string(OBFrameMetadataType type);
75
82 OB_EXPORT OBStreamType ob_sensor_type_to_stream_type(OBSensorType type);
83
90 OB_EXPORT const char *ob_format_to_string(OBFormat format);
91 #ifdef __cplusplus
92 }
93#endif
```

# TypeHelper.hpp File Reference

```
#include <string>
#include <iostream>
#include "libobsensor/h/ObTypes.h"
#include "libobsensor/h/TypeHelper.h"
#include <functional>
```

[Go to the source code of this file.](#)

## Classes

class **ob::TypeHelper**

## Namespaces

namespace **ob**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## TypeHelper.hpp

[Go to the documentation of this file.](#)

```

1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6 #include <string>
7 #include <iostream>
8 #include "libobssensor/h/ObTypes.h"
9 #include "libobssensor/h/TypeHelper.h"
10
11 #include <functional>
12
13 namespace ob {
14 class TypeHelper {
15 public:
16     static std::string convertOBFormatTypeToString(const OBFormat &type) {
17         return ob_format_type_to_string(type);
18     }
19
20     static std::string convertOBFrameTypeToString(const OBFrameType &type) {
21         return ob_frame_type_to_string(type);
22     }
23
24     static std::string convertOBStreamTypeToString(const OBStreamType &type) {
25         return ob_stream_type_to_string(type);
26     }
27
28     static std::string convertOBSSensorTypeToString(const OBSSensorType &type) {
29         return ob_sensor_type_to_string(type);
30     }
31
32     static std::string convertOBIMUSampleRateTypeToString(const OBIMUSampleRate &type) {
33         return ob_imu_rate_type_to_string(type);
34     }
35
36     static std::string convertOBGyroFullScaleRangeTypeToString(const OBGyroFullScaleRange &type) {
37         return ob_gyro_range_type_to_string(type);
38     }
39
40     static std::string convertOBAccelFullScaleRangeTypeToString(const OBAccelFullScaleRange &type) {
41         return ob_accel_range_type_to_string(type);
42     }
43
44     static std::string convertOBFrameMetadataTypeToString(const OBFrameMetadataType &type) {
45         return ob_meta_data_type_to_string(type);
46     }
47
48     static OBStreamType convertSensorTypeToStreamType(OBSSensorType type) {
49         return ob_sensor_type_to_stream_type(type);
50     }
51
52     static bool isVideoSensorType(OBSSensorType type) {
53         return ob_is_video_sensor_type(type);
54     }
55
56     static bool isVideoStreamType(OBStreamType type) {
57         return ob_is_video_stream_type(type);
58     }
59
60 };
61 } // namespace ob
62

```



# Types.hpp File Reference

```
#include <libobsensor/h/ObTypes.h>
```

Go to the source code of this file.

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Types.hpp

[Go to the documentation of this file.](#)

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.  
2 // Licensed under the MIT License.  
3  
4 #pragma once  
5 #include <libobssensor/h/ObTypes.h>
```

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Utils.h File Reference

```
#include "ObTypes.h"
```

Go to the source code of this file.

## Functions

```
OB_EXPORT bool ob_transformation_3d_to_3d(const OBPoint3f source_point3f,  
                                OBExtrinsic extrinsic, OBPoint3f *target_point3f, ob_error **error)
```

Transform a 3d point of a source coordinate system into a 3d point of the target coordinate system.

```
OB_EXPORT bool ob_transformation_2d_to_3d(const OBPoint2f source_point2f, const float  
                                         source_depth_pixel_value, const OBCameraIntrinsic source_intrinsic,  
                                         OBExtrinsic extrinsic, OBPoint3f *target_point3f, ob_error **error)
```

Transform a 2d pixel coordinate with an associated depth value of the source camera into a 3d point of the target coordinate system.

```
OB_EXPORT bool ob_transformation_3d_to_2d(const OBPoint3f source_point3f, const  
                                         OBCameraIntrinsic target_intrinsic, const OBCameraDistortion  
                                         target_distortion, OBExtrinsic extrinsic, OBPoint2f *target_point2f,  
                                         ob_error **error)
```

Transform a 3d point of a source coordinate system into a 2d pixel coordinate of the target camera.

```
OB_EXPORT bool ob_transformation_2d_to_2d(const OBPoint2f source_point2f, const float  
                                         source_depth_pixel_value, const OBCameraIntrinsic source_intrinsic, const  
                                         OBCameraDistortion source_distortion, const OBCameraIntrinsic  
                                         target_intrinsic, const OBCameraDistortion target_distortion, OBExtrinsic  
                                         extrinsic, OBPoint2f *target_point2f, ob_error **error)
```

Transform a 2d pixel coordinate with an associated depth value of the source camera into a 2d pixel coordinate of the target camera.

```
OB_EXPORT ob_frame * transformation_depth_frame_to_color_camera(ob_device *device,  
                                         ob_frame *depth_frame, uint32_t target_color_camera_width, uint32_t  
                                         target_color_camera_height, ob_error **error)
```

```
OB_EXPORT bool transformation_init_xy_tables(const ob_calibration_param  
                                         calibration_param, const ob_sensor_type sensor_type, float *data, uint32_t  
                                         *data_size, ob_xy_tables *xy_tables, ob_error **error)
```

```
OB_EXPORT void transformation_depth_to_pointcloud(ob_xy_tables *xy_tables, const void  
                                         *depth_image_data, void *pointcloud_data, ob_error **error)
```

```
OB_EXPORT void transformation_depth_to_rgbd_pointcloud(ob_xy_tables *xy_tables,  
                                         const void *depth_image_data, const void *color_image_data, void  
                                         *pointcloud_data, ob_error **error)
```

```

OB_EXPORT bool ob_calibration_3d_to_3d(const ob_calibration_param calibration_param,
    const ob_point3f source_point3f, const ob_sensor_type
    source_sensor_type, const ob_sensor_type target_sensor_type, ob_point3f
    *target_point3f, ob_error **error)

OB_EXPORT bool ob_calibration_2d_to_3d(const ob_calibration_param calibration_param,
    const ob_point2f source_point2f, const float source_depth_pixel_value, const
    ob_sensor_type source_sensor_type, const ob_sensor_type
    target_sensor_type, ob_point3f *target_point3f, ob_error **error)

OB_EXPORT bool ob_calibration_3d_to_2d(const ob_calibration_param calibration_param,
    const ob_point3f source_point3f, const ob_sensor_type
    source_sensor_type, const ob_sensor_type target_sensor_type, ob_point2f
    *target_point2f, ob_error **error)

OB_EXPORT bool ob_calibration_2d_to_2d(const ob_calibration_param calibration_param,
    const ob_point2f source_point2f, const float source_depth_pixel_value, const
    ob_sensor_type source_sensor_type, const ob_sensor_type
    target_sensor_type, ob_point2f *target_point2f, ob_error **error)

OB_EXPORT bool ob_save_pointcloud_to_ply(const char *file_name, ob_frame *frame, bool
    save_binary, bool use_mesh, float mesh_threshold, ob_error **error)
        save point cloud to ply file.

```

## Function Documentation

- ◆ **ob\_transformation\_3d\_to\_3d()**

```
OB_EXPORT bool ob_transformation_3d_to_3d ( const OBPoint3f source_point3f,  
                                              OBExtrinsic      extrinsic,  
                                              OBPoint3f *      target_point3f,  
                                              ob_error **     error )
```

Transform a 3d point of a source coordinate system into a 3d point of the target coordinate system.

### Parameters

[in] **source\_point3f** Source 3d point value  
[in] **extrinsic** Transformation matrix from source to target  
[out] **target\_point3f** Target 3d point value  
[out] **error** Pointer to an error object that will be set if an error occurs.

### Returns

bool Transform result

Referenced by [ob::CoordinateTransformHelper::transformation3dto3d\(\)](#).

- ◆ [ob\\_transformation\\_2d\\_to\\_3d\(\)](#)

```
OB_EXPORT bool ob_transformation_2d_to_3d ( const OBPoint2f source_point2f,
                                             const float source_depth_pixel_value,
                                             const OBCameraIntrinsic source_intrinsic,
                                             OBExtrinsic extrinsic,
                                             OBPoint3f * target_point3f,
                                             ob_error ** error )
```

Transform a 2d pixel coordinate with an associated depth value of the source camera into a 3d point of the target coordinate system.

### Parameters

[in] <b>source_point2f</b>	Source 2d point value
[in] <b>source_depth_pixel_value</b>	The depth of sourcePoint2f in millimeters
[in] <b>source_intrinsic</b>	Source intrinsic parameters
[in] <b>extrinsic</b>	Transformation matrix from source to target
[out] <b>target_point3f</b>	Target 3d point value
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

bool Transform result

Referenced by **ob::CoordinateTransformHelper::transformation2dto3d()**.

◆ **ob\_transformation\_3d\_to\_2d()**

```
OB_EXPORT bool ob_transformation_3d_to_2d ( const OBPoint3f source_point3f,  
                                              const OBCameraIntrinsic target_intrinsic,  
                                              const OBCameraDistortion target_distortion,  
                                              OBExtrinsic extrinsic,  
                                              OBPoint2f * target_point2f,  
                                              ob_error ** error )
```

Transform a 3d point of a source coordinate system into a 2d pixel coordinate of the target camera.

### Parameters

[in] <b>source_point3f</b>	Source 3d point value
[in] <b>target_intrinsic</b>	Target intrinsic parameters
[in] <b>target_distortion</b>	Target distortion parameters
[in] <b>extrinsic</b>	Transformation matrix from source to target
[out] <b>target_point2f</b>	Target 2d point value
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

**bool** Transform result

Referenced by [ob::CoordinateTransformHelper::transformation3dto2d\(\)](#).

◆ [ob\\_transformation\\_2d\\_to\\_2d\(\)](#)

```
OB_EXPORT bool ob_transformation_2d_to_2d ( const OBPoint2f source_point2f,
                                             const float source_depth_pixel_value,
                                             const OBCameraIntrinsic source_intrinsic,
                                             const OBCameraDistortion source_distortion,
                                             const OBCameraIntrinsic target_intrinsic,
                                             const OBCameraDistortion target_distortion,
                                             OBExtrinsic extrinsic,
                                             OBPoint2f * target_point2f,
                                             ob_error ** error )
```

Transform a 2d pixel coordinate with an associated depth value of the source camera into a 2d pixel coordinate of the target camera.

## Parameters

[in] <b>source_intrinsic</b>	Source intrinsic parameters
[in] <b>source_distortion</b>	Source distortion parameters
[in] <b>source_point2f</b>	Source 2d point value
[in] <b>source_depth_pixel_value</b>	The depth of sourcePoint2f in millimeters
[in] <b>target_intrinsic</b>	Target intrinsic parameters
[in] <b>target_distortion</b>	Target distortion parameters
[in] <b>extrinsic</b>	Transformation matrix from source to target
[out] <b>target_point2f</b>	Target 2d point value
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

## Returns

bool Transform result

Referenced by [ob::CoordinateTransformHelper::transformation2dto2d\(\)](#).

- ◆ [transformation\\_depth\\_frame\\_to\\_color\\_camera\(\)](#)

```
OB_EXPORT ob_frame *  
transformation_depth_frame_to_color_camera  
(ob_device * device,  
ob_frame * depth_frame,  
uint32_t target_color_camera_width,  
uint32_t target_color_camera_height,  
ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::transformationDepthFrameToColorCamera\(\)](#).

◆ [transformation\\_init\\_xy\\_tables\(\)](#)

```
OB_EXPORT bool transformation_init_xy_tables ( const ob_calibration_param calibration_param,  
                                              const ob_sensor_type sensor_type,  
                                              float * data,  
                                              uint32_t * data_size,  
                                              ob_xy_tables * xy_tables,  
                                              ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::transformationInitXYTables\(\)](#).

◆ [transformation\\_depth\\_to\\_pointcloud\(\)](#)

```
OB_EXPORT void transformation_depth_to_pointcloud ( ob_xy_tables * xy_tables,  
                                              const void * depth_image_data,  
                                              void * pointcloud_data,  
                                              ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::transformationDepthToPointCloud\(\)](#).

◆ [transformation\\_depth\\_to\\_rgbd\\_pointcloud\(\)](#)

```
OB_EXPORT void transformation_depth_to_rgbd_pointcloud ( ob_xy_tables * xy_tables,  
                                              const void * depth_image_data,  
                                              const void * color_image_data,  
                                              void * pointcloud_data,  
                                              ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::transformationDepthToRGBDPointCloud\(\)](#).

◆ [ob\\_calibration\\_3d\\_to\\_3d\(\)](#)

```
OB_EXPORT bool ob_calibration_3d_to_3d ( const ob_calibration_param calibration_param,  
                                         const ob_point3f source_point3f,  
                                         const ob_sensor_type source_sensor_type,  
                                         const ob_sensor_type target_sensor_type,  
                                         ob_point3f * target_point3f,  
                                         ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::calibration3dTo3d\(\)](#).

◆ [ob\\_calibration\\_2d\\_to\\_3d\(\)](#)

```
OB_EXPORT bool ob_calibration_2d_to_3d ( const ob_calibration_param calibration_param,  
                                         const ob_point2f source_point2f,  
                                         const float source_depth_pixel_value,  
                                         const ob_sensor_type source_sensor_type,  
                                         const ob_sensor_type target_sensor_type,  
                                         ob_point3f * target_point3f,  
                                         ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::calibration2dTo3d\(\)](#).

◆ [ob\\_calibration\\_3d\\_to\\_2d\(\)](#)

```
OB_EXPORT bool ob_calibration_3d_to_2d ( const ob_calibration_param calibration_param,  
                                         const ob_point3f source_point3f,  
                                         const ob_sensor_type source_sensor_type,  
                                         const ob_sensor_type target_sensor_type,  
                                         ob_point2f* target_point2f,  
                                         ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::calibration3dTo2d\(\)](#).

◆ [ob\\_calibration\\_2d\\_to\\_2d\(\)](#)

```
OB_EXPORT bool ob_calibration_2d_to_2d ( const ob_calibration_param calibration_param,  
                                         const ob_point2f source_point2f,  
                                         const float source_depth_pixel_value,  
                                         const ob_sensor_type source_sensor_type,  
                                         const ob_sensor_type target_sensor_type,  
                                         ob_point2f* target_point2f,  
                                         ob_error ** error )
```

Referenced by [ob::CoordinateTransformHelper::calibration2dTo2d\(\)](#).

◆ [ob\\_save\\_pointcloud\\_to\\_ply\(\)](#)

```
OB_EXPORT bool ob_save_pointcloud_to_ply ( const char * file_name,  
                                         ob_frame * frame,  
                                         bool      save_binary,  
                                         bool      use_mesh,  
                                         float     mesh_threshold,  
                                         ob_error ** error )
```

save point cloud to ply file.

### Parameters

[in] <b>file_name</b>	Point cloud save path
[in] <b>frame</b>	Point cloud frame
[in] <b>save_binary</b>	Binary or textual,true: binary, false: textual
[in] <b>use_mesh</b>	Save mesh or not, true: save as mesh, false: not save as mesh
[in] <b>mesh_threshold</b>	Distance threshold for creating faces in point cloud,default value :50
[out] <b>error</b>	Pointer to an error object that will be set if an error occurs.

### Returns

bool save point cloud result

Referenced by [ob::PointCloudHelper::savePointcloudToPly\(\)](#).

## Utils.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5
6 #ifndef __cplusplus
7 extern "C" {
8 #endif
9
10 #include "ObTypes.h"
11
12 OB_EXPORT bool ob_transformation_3d_to_3d(const OBPoint3f source_point3f, OBExtrinsic extrinsic, O
    BPoint3f*target_point3f, ob_error **error);
13
14 OB_EXPORT bool ob_transformation_2d_to_3d(const OBPoint2f source_point2f, const float source_dept
    h_pixel_value, const OBCameraIntrinsic source_intrinsic,
    OBExtrinsic extrinsic, OBPoint3f*target_point3f, ob_error **error);
15
16 OB_EXPORT bool ob_transformation_3d_to_2d(const OBPoint3f source_point3f, const OBCameraIntrinsic
    target_intrinsic, const OBCameraDistortion target_distortion,
    OBExtrinsic extrinsic, OBPoint2f*target_point2f, ob_error **error);
17
18 OB_EXPORT bool ob_transformation_2d_to_2d(const OBPoint2f source_point2f, const float source_dept
    h_pixel_value, const OBCameraIntrinsic source_intrinsic,
    const OBCameraDistortion source_distortion, const OBCameraIntrinsic target_intri
    ntic,
    const OBCameraDistortion target_distortion, OBExtrinsic extrinsic, OBPoint2f*targ
    et_point2f, ob_error **error);
19
20 //\deprecated This function is deprecated and will be removed in a future version.
21 OB_EXPORT ob_frame *transformation_depth_frame_to_color_camera(ob_device *device, ob_frame *de
    pth_frame, uint32_t target_color_camera_width,
    uint32_t target_color_camera_height, ob_error **error);
22
23 //\deprecated This function is deprecated and will be removed in a future version.
24 OB_EXPORT bool transformation_init_xy_tables(const ob_calibration_param calibration_param, const ob
    _sensor_type sensor_type, float *data, uint32_t *data_size,
    ob_xy_tables *xy_tables, ob_error **error);
25
26 //\deprecated This function is deprecated and will be removed in a future version.
27 OB_EXPORT void transformation_depth_to_pointcloud(ob_xy_tables *xy_tables, const void *depth_im
    age_data, void *pointcloud_data, ob_error **error);
28
29 //\deprecated This function is deprecated and will be removed in a future version.
30 OB_EXPORT void transformation_depth_to_rgbd_pointcloud(ob_xy_tables *xy_tables, const void *dept
    h_image_data, const void *color_image_data,
    void *pointcloud_data, ob_error **error);
31
32 //\deprecated This function is deprecated and will be removed in a future version.
33 //      Use the ob_transformation_3d_to_3d instead.
34 OB_EXPORT bool ob_calibration_3d_to_3d(const ob_calibration_param calibration_param, const ob poi
    nt3f source_point3f, const ob_sensor_type source_sensor_type,
    const ob_sensor_type target_sensor_type, ob_point3f*target_point3f, ob_error **
    error);
35
36 //\deprecated This function is deprecated and will be removed in a future version.
37 //      Use the ob_transformation_2d_to_3d instead.
38 OB_EXPORT bool ob_calibration_2d_to_3d(const ob_calibration_param calibration_param, const ob poi
```

```

96     nt2f source_point2f, const float source_depth_pixel_value,
97         const ob_sensor_type source_sensor_type, const ob_sensor_type target_sensor_t
98             ype, ob_point3f*target_point3f,
99                 ob_error **error);
100
101 // \deprecated This function is deprecated and will be removed in a future version.
102 //       Use the ob_transformation_3d_to_2d instead.
103 OB_EXPORT bool ob_calibration_3d_to_2d(const ob_calibration_param calibration_param, const ob_poi
104 nt3f source_point3f, const ob_sensor_type source_sensor_type,
105                 const ob_sensor_type target_sensor_type, ob_point2f*target_point2f, ob_error **
106 error);
107
108 // \deprecated This function is deprecated and will be removed in a future version.
109 //       Use the ob_transformation_2d_to_2d instead.
110 OB_EXPORT bool ob_calibration_2d_to_2d(const ob_calibration_param calibration_param, const ob_poi
111 nt2f source_point2f, const float source_depth_pixel_value,
112                 const ob_sensor_type source_sensor_type, const ob_sensor_type target_sensor_t
113             ype, ob_point2f*target_point2f,
114                 ob_error **error);
115 OB_EXPORT bool ob_save_pointcloud_to_ply(const char *file_name, ob_frame *frame, bool save_binar
116 y, bool use_mesh, float mesh_threshold, ob_error **error);
117 #ifndef __cplusplus
118 }
119 #endif

```

# Utils.hpp File Reference

The SDK utils class. [More...](#)

```
#include "libobsensor/h/Utils.h"  
#include "Device.hpp"  
#include "Types.hpp"  
#include "Frame.hpp"  
#include <memory>
```

[Go to the source code of this file.](#)

## Classes

```
class ob::CoordinateTransformHelper  
class ob::PointCloudHelper
```

## Namespaces

```
namespace ob
```

## Detailed Description

The SDK utils class.

Definition in file [Utils.hpp](#).

# Utils.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbee Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
4 #pragma once
5 #include "libobsensor/h/Utils.h"
6 #include "Device.hpp"
7 #include "Types.hpp"
8 #include "Frame.hpp"
9
10 #include <memory>
11
12 namespace ob {
13     class Device;
14
15     class CoordinateTransformHelper {
16     public:
17         static bool transformation3dto3d(const OBPoint3f source_point3f, OBExtrinsic extrinsic, OBPoint3f *target_point3f) {
18             ob_error *error = NULL;
19             bool result = ob_transformation_3d_to_3d(source_point3f, extrinsic, target_point3f, &error);
20             Error::handle(&error);
21             return result;
22         }
23
24         static bool transformation2dto3d(const OBPoint2f source_point2f, const float source_depth_pixel_value,
25                                         const OBCameraIntrinsic source_intrinsic,
26                                         OBExtrinsic extrinsic, OBPoint3f *target_point3f) {
27             ob_error *error = NULL;
28             bool result = ob_transformation_2d_to_3d(source_point2f, source_depth_pixel_value, source_intrinsic, extrinsic, target_point3f, &error);
29             Error::handle(&error);
30             return result;
31         }
32
33         static bool transformation3dto2d(const OBPoint3f source_point3f, const OBCameraIntrinsic target_intrinsic,
34                                         const OBCameraDistortion target_distortion,
35                                         OBExtrinsic extrinsic, OBPoint2f *target_point2f) {
36             ob_error *error = NULL;
37             bool result = ob_transformation_3d_to_2d(source_point3f, target_intrinsic, target_distortion, extrinsic, target_point2f, &error);
38             Error::handle(&error);
39             return result;
40         }
41
42         static bool transformation2dto2d(const OBPoint2f source_point2f, const float source_depth_pixel_value,
43                                         const OBCameraIntrinsic source_intrinsic,
44                                         const OBCameraDistortion source_distortion, const OBCameraIntrinsic target_intrinsic,
45                                         const OBCameraDistortion target_distortion, OBExtrinsic extrinsic, OBPoint2f *target_point2f) {
46             ob_error *error = NULL;
47             bool result = ob_transformation_2d_to_2d(source_point2f, source_depth_pixel_value, source_intrinsic, source_distortion, target_intrinsic,
48                                         target_distortion, extrinsic, target_point2f, &error);
49             Error::handle(&error);
50             return result;
51         }
52     };
53 }
```

```

100 public:
101     // The following interfaces are deprecated and are retained here for compatibility purposes.
102     static bool calibration3dTo3d(const OBCalibrationParam calibrationParam, const OBPoint3f sourcePoint
103         3f, const OBSensorType sourceSensorType,
104             const OBSensorType targetSensorType, OBPoint3f *targetPoint3f) {
105         ob_error *error = NULL;
106         bool result = ob_calibration_3d_to_3d(calibrationParam, sourcePoint3f, sourceSensorType, targetS
107             ensorType, targetPoint3f, &error);
108         Error::handle(&error);
109         return result;
110     }
111
110 static bool calibration2dTo3d(const OBCalibrationParam calibrationParam, const OBPoint2f sourcePoint
111     2f, const float sourceDepthPixelValue,
112         const OBSensorType sourceSensorType, const OBSensorType targetSensorType, OB
113             Point3f *targetPoint3f) {
114         ob_error *error = NULL;
115         bool result =
116             ob_calibration_2d_to_3d(calibrationParam, sourcePoint2f, sourceDepthPixelValue, sourceSensorTy
117                 pe, targetSensorType, targetPoint3f, &error);
118         Error::handle(&error);
119         return result;
120     }
121
120 static bool calibration3dTo2d(const OBCalibrationParam calibrationParam, const OBPoint3f sourcePoint
121     3f, const OBSensorType sourceSensorType,
122         const OBSensorType targetSensorType, OBPoint2f *targetPoint2f) {
123         ob_error *error = NULL;
124         bool result = ob_calibration_3d_to_2d(calibrationParam, sourcePoint3f, sourceSensorType, targetS
125             ensorType, targetPoint2f, &error);
126         Error::handle(&error);
127         return result;
128     }
129
127 static bool calibration2dTo2d(const OBCalibrationParam calibrationParam, const OBPoint2f sourcePoint
128     2f, const float sourceDepthPixelValue,
129         const OBSensorType sourceSensorType, const OBSensorType targetSensorType, OB
130             Point2f *targetPoint2f) {
131         ob_error *error = NULL;
132         bool result =
133             ob_calibration_2d_to_2d(calibrationParam, sourcePoint2f, sourceDepthPixelValue, sourceSensorTy
134                 pe, targetSensorType, targetPoint2f, &error);
135         Error::handle(&error);
136         return result;
137     }
138
136 static std::shared_ptr<ob::Frame> transformationDepthFrameToColorCamera(std::shared_ptr<ob::Devic
137     e> device, std::shared_ptr<ob::Frame> depthFrame,
138                     uint32_t targetColorCameraWidth, uint32_t targetColorCamera
139                     Height) {
140         ob_error *error = NULL;
141
140         // unsafe operation, need to cast const to non-const
141         auto unConstImpl = const_cast<ob_frame *>(depthFrame->getImpl());
142
143         auto result = transformation_depth_frame_to_color_camera(device->getImpl(), unConstImpl, targetCo
144             lorCameraWidth, targetColorCameraHeight, &error);
145         Error::handle(&error);
146         return std::make_shared<ob::Frame>(result);
147     }
148
148 static bool transformationInitXYTables(const OBCalibrationParam calibrationParam, const OBSensorTy
149     pe sensorType, float *data, uint32_t *dataSize,
150             OBXYTables *xyTables) {
150         ob_error *error = NULL;

```

```

151     bool result = transformation_init_xy_tables(calibrationParam, sensorType, data, dataSize, xyTables,
152                                                 &error);
153     Error::handle(&error);
154     return result;
155 }
156
156 static void transformationDepthToPointCloud(OBXYTables *xyTables, const void *depthImageData, v
157     oid *pointCloudData) {
158     ob_error *error = NULL;
159     transformation_depth_to_pointcloud(xyTables, depthImageData, pointCloudData, &error);
160     Error::handle(&error, false);
161 }
162
162 static void transformationDepthToRGBDPointCloud(OBXYTables *xyTables, const void *depthImageD
163     ata, const void *colorImageData, void *pointCloudData) {
164     ob_error *error = NULL;
165     transformation_depth_to_rgbd_pointcloud(xyTables, depthImageData, colorImageData, pointCloudD
166     ata, &error);
167     Error::handle(&error, false);
168 }
169
169 class PointCloudHelper {
170 public:
182     static bool savePointCloudToPly(const char *fileName, std::shared_ptr<ob::Frame> frame, bool saveBin
183         ary, bool useMesh, float meshThreshold) {
184         ob_error *error = NULL;
185         auto unConstImpl = const_cast<ob::frame*>(frame->getImpl());
186         bool result = ob_save_pointcloud_to_ply(fileName, unConstImpl, saveBinary, useMesh, meshT
187         hreshold, &error);
188         Error::handle(&error, false);
189         return result;
190     }
190 };
190 } // namespace ob

```

# Version.h File Reference

Functions for retrieving the SDK version number information. [More...](#)

```
#include "Export.h"
```

Go to the source code of this file.

## Functions

**OB\_EXPORT** int **ob\_get\_version()**

Get the SDK version number.

**OB\_EXPORT** int **ob\_get\_major\_version()**

Get the SDK major version number.

**OB\_EXPORT** int **ob\_get\_minor\_version()**

Get the SDK minor version number.

**OB\_EXPORT** int **ob\_get\_patch\_version()**

Get the SDK patch version number.

**OB\_EXPORT** const char \* **ob\_get\_stage\_version()**

Get the SDK stage version.

## Detailed Description

Functions for retrieving the SDK version number information.

Definition in file **Version.h**.

## Function Documentation

### ◆ **ob\_get\_version()**

**OB\_EXPORT** int ob\_get\_version( )

Get the SDK version number.

#### Returns

int The SDK version number.

Referenced by **ob::Version::getVersion()**.

◆ **ob\_get\_major\_version()**

**OB\_EXPORT** int ob\_get\_major\_version ( )

Get the SDK major version number.

**Returns**

int The SDK major version number.

Referenced by **ob::Version::getMajor()**.

◆ **ob\_get\_minor\_version()**

**OB\_EXPORT** int ob\_get\_minor\_version ( )

Get the SDK minor version number.

**Returns**

int The SDK minor version number.

Referenced by **ob::Version::getMinor()**.

◆ **ob\_get\_patch\_version()**

**OB\_EXPORT** int ob\_get\_patch\_version ( )

Get the SDK patch version number.

**Returns**

int The SDK patch version number.

Referenced by **ob::Version::getPatch()**.

◆ **ob\_get\_stage\_version()**

**OB\_EXPORT** const char\* ob\_get\_stage\_version( )

Get the SDK stage version.

### Attention

The returned char\* does not need to be freed.

### Returns

const char\* The SDK stage version.

Referenced by [ob::Version::getStageVersion\(\)](#).

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# Version.h

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
9 #pragma once
10
11 #include "Export.h"
12
13 #ifdef __cplusplus
14 extern "C" {
15 #endif
16
22 OB_EXPORT int ob_get_version();
23
29 OB_EXPORT int ob_get_major_version();
30
36 OB_EXPORT int ob_get_minor_version();
37
43 OB_EXPORT int ob_get_patch_version();
44
51 OB_EXPORT const char *ob_get_stage_version();
52
53 #ifndef __cplusplus
54 }
55 #endif
```

# Version.hpp File Reference

Provides functions to retrieve version information of the SDK. [More...](#)

```
#include "libobsensor/h/Version.h"
```

[Go to the source code of this file.](#)

## Classes

```
class ob::Version
```

## Namespaces

```
namespace ob
```

## Detailed Description

Provides functions to retrieve version information of the SDK.

Definition in file [Version.hpp](#).

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Version.hpp

Go to the documentation of this file.

```
1 // Copyright (c) Orbbec Inc. All Rights Reserved.
2 // Licensed under the MIT License.
3
8 #pragma once
9
10 #include "libobsensor/h/Version.h"
11
12 namespace ob {
13 class Version {
14 public:
21     static int getVersion() {
22         return ob_get_version();
23     }
24
29     static int getMajor() {
30         return ob_get_major_version();
31     }
32
38     static int getMinor() {
39         return ob_get_minor_version();
40     }
41
47     static int getPatch() {
48         return ob_get_patch_version();
49     }
50
57     static const char *getStageVersion() {
58         return ob_get_stage_version();
59     }
60 };
61 } // namespace ob
62
```

# Class List

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

[detail level 1 2 ]

N <a href="#">ob</a>	
C <a href="#">AE_ROI</a>	The rect of the region of interest
C <a href="#">BASELINE_CALIBRATION_PARAM</a>	Baseline calibration parameters
C <a href="#">HDR_CONFIG</a>	HDR Configuration
C <a href="#">ob_device_timestamp_reset_config</a>	The timestamp reset configuration of the device
C <a href="#">ob_error</a>	The error class exposed by the SDK, users can get detailed error information according to the error
C <a href="#">ob_margin_filter_config</a>	Configuration for depth margin filter
C <a href="#">ob_multi_device_sync_config</a>	The synchronization configuration of the device
C <a href="#">OBAccelIntrinsic</a>	Structure for accelerometer intrinsic parameters
C <a href="#">OBAccelValue</a>	Data structures for accelerometers and gyroscopes
C <a href="#">OBBoolPropertyRange</a>	Structure for boolean range
C <a href="#">OBCalibrationParam</a>	Calibration parameters
C <a href="#">OBCameraDistortion</a>	Structure for distortion parameters
C <a href="#">OBCameraIntrinsic</a>	Structure for camera intrinsic parameters
C <a href="#">OBCameraParam</a>	Structure for camera parameters
C <a href="#">OBColorPoint</a>	3D point structure with color information
C <a href="#">OBCompressionParams</a>	
C <a href="#">OBD2CTransform</a>	Structure for rotation/transformation
C <a href="#">OBDataChunk</a>	Structure for transmitting data blocks
C <a href="#">OBDepthWorkMode</a>	Depth work mode
C <a href="#">OBDeviceSerialNumber</a>	Struct of serial number
C <a href="#">OBDeviceSyncConfig</a>	Device synchronization configuration
C <a href="#">OBDeviceTemperature</a>	Temperature parameters of the device (unit: Celsius)
C <a href="#">OBDisparityParam</a>	Disparity parameters for disparity based camera
C <a href="#">OBDispOffsetConfig</a>	Disparity offset interleaving configuration
C <a href="#">OBEdgeNoiseRemovalFilterParams</a>	
C <a href="#">OBFilterConfigSchemaItem</a>	Configuration Item for the filter
C <a href="#">OBFloatPropertyRange</a>	Structure for float range

<b>C OBGyroIntrinsic</b>	Structure for gyroscope intrinsic parameters
<b>C OBIntPropertyRange</b>	Structure for integer range
<b>C OBMGCFilterConfig</b>	Configuration for mgc filter
<b>C OBNetIpConfig</b>	IP address configuration for network devices (IPv4)
<b>C OBNoiseRemovalFilterParams</b>	
<b>C OBPoint</b>	3D point structure in the SDK
<b>C OBPoint2f</b>	2D point structure in the SDK
<b>C OBPresetResolutionConfig</b>	
<b>C OBPropertyItem</b>	Used to describe the characteristics of each property
<b>C OBProtocolVersion</b>	Control command protocol version number
<b>C OBRect</b>	Rectangle
<b>C OBSequenceIdItem</b>	SequenceId filter list item
<b>C OBSpatialAdvancedFilterParams</b>	
<b>C OBTofExposureThresholdControl</b>	TOF Exposure Threshold
<b>C OBUInt16PropertyRange</b>	Structure for float range
<b>C OBUInt8PropertyRange</b>	Structure for float range
<b>C OBXYTables</b>	

# Class Index

A | B | C | D | E | F | G | H | I | N | O | P | R | S | T | V

## A

[AccelFrame \(ob\)](#)  
[AccelStreamProfile \(ob\)](#)  
[AE\\_ROI](#)  
[Align \(ob\)](#)

## B

[BASELINE\\_CALIBRATION\\_PARAM](#)

## C

[CameraParamList \(ob\)](#)  
[ColorFrame \(ob\)](#)  
[ConfidenceFrame \(ob\)](#)  
[Config \(ob\)](#)  
[Context \(ob\)](#)  
[CoordinateTransformHelper \(ob\)](#)

## D

[DecimationFilter \(ob\)](#)  
[DepthFrame \(ob\)](#)  
[Device \(ob\)](#)  
[DeviceFrameInterleaveList \(ob\)](#)  
[DeviceInfo \(ob\)](#)  
[DeviceList \(ob\)](#)  
[DevicePresetList \(ob\)](#)  
[DisparityTransform \(ob\)](#)

## E

[Error \(ob\)](#)

## F

[Filter \(ob\)](#)  
[FilterFactory \(ob\)](#)  
[FormatConvertFilter \(ob\)](#)

**Frame (ob)**  
**FrameFactory (ob)**  
**FrameHelper (ob)**  
**FrameSet (ob)**

## G

**GyroFrame (ob)**  
**GyroStreamProfile (ob)**

## H

**HDR\_CONFIG**  
**HdrMerge (ob)**  
**HoleFillingFilter (ob)**

## I

**IRFrame (ob)**

## N

**NoiseRemovalFilter (ob)**

## O

**ob\_device\_timestamp\_reset\_config**  
**ob\_error**  
**ob\_margin\_filter\_config**  
**ob\_multi\_device\_sync\_config**  
**OBAccelIntrinsic**  
**OBAccelValue**  
**OBBoolPropertyRange**  
**OBCalibrationParam**  
**OBCameraDistortion**  
**OBCameraIntrinsic**  
**OBCameraParam**  
**OBColorPoint**  
**OBCompressionParams**  
**OBD2CTransform**  
**OBDataChunk**  
**OBDepthWorkMode**  
**OBDepthWorkModeList (ob)**  
**OBDeviceSerialNumber**  
**OBDeviceSyncConfig**  
**OBDeviceTemperature**  
**OBDisparityParam**  
**OBDispOffsetConfig**

[OBEdgeNoiseRemovalFilterParams](#)  
[OBFilterConfigSchemaItem](#)  
[OBFilterList \(ob\)](#)  
[OBFloatPropertyRange](#)  
[OBGyroIntrinsic](#)  
[OBIntPropertyRange](#)  
[OBMGCFilterConfig](#)  
[OBNetIpConfig](#)  
[OBNoiseRemovalFilterParams](#)  
[OBPoint](#)  
[OBPoint2f](#)  
[OBPresetResolutionConfig](#)  
[OBPropertyItem](#)  
[OBProtocolVersion](#)  
[OBRect](#)  
[OBSequenceIdItem](#)  
[OBSpatialAdvancedFilterParams](#)  
[OBTofExposureThresholdControl](#)  
[OBUInt16PropertyRange](#)  
[OBUInt8PropertyRange](#)  
[OBXYTables](#)

## P

[Pipeline \(ob\)](#)  
[PlaybackDevice \(ob\)](#)  
[PointCloudFilter \(ob\)](#)  
[PointCloudHelper \(ob\)](#)  
[PointsFrame \(ob\)](#)  
[PresetResolutionConfigList \(ob\)](#)

## R

[RangeTraits \(ob\)](#)  
[RangeTraits< OBFloatPropertyRange > \(ob\)](#)  
[RangeTraits< OBIntPropertyRange > \(ob\)](#)  
[RangeTraits< OBUInt16PropertyRange > \(ob\)](#)  
[RangeTraits< OBUInt8PropertyRange > \(ob\)](#)  
[RecordDevice \(ob\)](#)

## S

[Sensor \(ob\)](#)  
[SensorList \(ob\)](#)  
[SequenceIdFilter \(ob\)](#)  
[SpatialAdvancedFilter \(ob\)](#)  
[StreamProfile \(ob\)](#)  
[StreamProfileFactory \(ob\)](#)  
[StreamProfileList \(ob\)](#)

# T

[\*\*TemporalFilter \(ob\)\*\*](#)  
[\*\*ThresholdFilter \(ob\)\*\*](#)  
[\*\*TypeHelper \(ob\)\*\*](#)

# V

[\*\*Version \(ob\)\*\*](#)  
[\*\*VideoFrame \(ob\)\*\*](#)  
[\*\*VideoStreamProfile \(ob\)\*\*](#)

## ob::AccelFrame Member List

This is the complete list of members for **ob::AccelFrame**, including all inherited members.

<b>AccelFrame</b> (const ob_frame *impl)	<b>ob::AccelFrame</b>	inline explicit
<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame</b> (const ob_frame *impl)	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTemperature() const</b>	<b>ob::AccelFrame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getValue() const</b>	<b>ob::AccelFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	
<b>metadata() const</b>	<b>ob::Frame</b>	inline
<b>metadataSize() const</b>	<b>ob::Frame</b>	inline
<b>systemTimeStamp() const</b>	<b>ob::Frame</b>	inline

<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>temperature()</b>	<b>ob::AccelFrame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>value()</b>	<b>ob::AccelFrame</b>	inline
<b>~AccelFrame()</b> noexcept override=default	<b>ob::AccelFrame</b>	
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::AccelFrame Class Reference

Define the **AccelFrame** class, which inherits from the **Frame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::AccelFrame:

## Public Member Functions

```
AccelFrame (const ob_frame *impl)  
~AccelFrame () noexcept override=default
```

```
OBAccelValue getValue () const
```

Get the accelerometer frame data.

```
float getTemperature () const
```

Get the temperature when the frame was sampled.

```
OBAccelValue value ()
```

```
float temperature ()
```

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **AccelFrame** class, which inherits from the **Frame** class.

Definition at line **654** of file **Frame.hpp**.

## Constructor & Destructor Documentation

◆ **AccelFrame()**

```
ob::AccelFrame::AccelFrame ( const ob_frame * impl )
```

inline explicit

Definition at line **657** of file **Frame.hpp**.

◆ ~AccelFrame()

ob::AccelFrame::~AccelFrame ( )

override default noexcept

## Member Function Documentation

◆ getValue()

**OBAccelValue** ob::AccelFrame::getValue ( ) const

inline

Get the accelerometer frame data.

**Returns**

**OBAccelValue** The accelerometer frame data

Definition at line **666** of file **Frame.hpp**.

Referenced by [getValue\(\)](#), and [value\(\)](#).

◆ getTemperature()

float ob::AccelFrame::getTemperature ( ) const

inline

Get the temperature when the frame was sampled.

**Returns**

float The temperature value in celsius

Definition at line **679** of file **Frame.hpp**.

Referenced by [temperature\(\)](#).

◆ value()

**OBAccelValue** ob::AccelFrame::value ( )

inline

Definition at line **689** of file **Frame.hpp**.

Referenced by [getValue\(\)](#).

◆ **temperature()**

float ob::AccelFrame::temperature ( )

inline

Definition at line **693** of file [Frame.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::AccelStreamProfile Member List

This is the complete list of members for **ob::AccelStreamProfile**, including all inherited members.

<b>AccelStreamProfile</b> (const ob_stream_profile_t *impl)	<b>ob::AccelStreamP</b>
<b>as()</b>	<b>ob::StreamProfile</b>
<b>as()</b> const	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo</b> (std::shared_ptr< StreamProfile > target, const OBExtrinsic &extrinsic)	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo</b> (const OBStreamType &targetStreamType, const OBExtrinsic &extrinsic)	<b>ob::StreamProfile</b>
<b>format()</b> const	<b>ob::StreamProfile</b>
<b>fullScaleRange()</b> const	<b>ob::AccelStreamP</b>
<b>getExtrinsicTo</b> (std::shared_ptr< StreamProfile > target) const	<b>ob::StreamProfile</b>
<b>getFormat()</b> const	<b>ob::StreamProfile</b>
<b>getFullScaleRange()</b> const	<b>ob::AccelStreamP</b>
<b>getImpl()</b> const	<b>ob::StreamProfile</b>
<b>getIntrinsic()</b> const	<b>ob::AccelStreamP</b>
<b>getSampleRate()</b> const	<b>ob::AccelStreamP</b>
<b>getType()</b> const	<b>ob::StreamProfile</b>
<b>impl_</b>	<b>ob::StreamProfile</b>
<b>is()</b> const	<b>ob::StreamProfile</b>
<b>operator=(</b> StreamProfile &streamProfile) <b>=delete</b>	<b>ob::StreamProfile</b>
<b>operator=(</b> StreamProfile &&streamProfile) noexcept	<b>ob::StreamProfile</b>
<b>sampleRate()</b> const	<b>ob::AccelStreamP</b>
<b>StreamProfile</b> (StreamProfile &streamProfile) <b>=delete</b>	<b>ob::StreamProfile</b>
<b>StreamProfile</b> (StreamProfile &&streamProfile) noexcept	<b>ob::StreamProfile</b>
<b>StreamProfile</b> (const ob_stream_profile_t *impl)	<b>ob::StreamProfile</b>
<b>type()</b> const	<b>ob::StreamProfile</b>
<b>~AccelStreamProfile()</b> noexcept override=default	<b>ob::AccelStreamP</b>
<b>~StreamProfile()</b> noexcept	<b>ob::StreamProfile</b>

# ob::AccelStreamProfile Class Reference

Class representing an accelerometer stream profile. [More...](#)

```
#include <StreamProfile.hpp>
```

Inheritance diagram for ob::AccelStreamProfile:

## Public Member Functions

**AccelStreamProfile** (const ob\_stream\_profile\_t \*impl)

**~AccelStreamProfile** () noexcept override=default

**OBAccelFullScaleRange** **getFullScaleRange** () const

Return the full scale range.

**OBAccelSampleRate** **getSampleRate** () const

Return the sampling frequency.

**OBAccelIntrinsic** **getIntrinsic** () const

get the intrinsic parameters of the stream.

**OBAccelFullScaleRange** **fullScaleRange** () const

**OBAccelSampleRate** **sampleRate** () const

Public Member Functions inherited from **ob::StreamProfile**

## Additional Inherited Members

Protected Member Functions inherited from **ob::StreamProfile**

Protected Attributes inherited from **ob::StreamProfile**

## Detailed Description

Class representing an accelerometer stream profile.

Definition at line **272** of file **StreamProfile.hpp**.

## Constructor & Destructor Documentation

- ◆ **AccelStreamProfile()**

```
ob::AccelStreamProfile::AccelStreamProfile ( const ob_stream_profile_t * impl )
```

inline explicit

Definition at line [274](#) of file [StreamProfile.hpp](#).

◆ ~AccelStreamProfile()

```
ob::AccelStreamProfile::~AccelStreamProfile ( )
```

override default noexcept

## Member Function Documentation

◆ getFullScaleRange()

**OBAccelFullScaleRange** ob::AccelStreamProfile::getFullScaleRange ( ) const

inline

Return the full scale range.

**Returns**

**OBAccelFullScaleRange** Return the scale range value.

Definition at line [283](#) of file [StreamProfile.hpp](#).

Referenced by [fullScaleRange\(\)](#), and [getFullScaleRange\(\)](#).

◆ getSampleRate()

**OBAccelSampleRate** ob::AccelStreamProfile::getSampleRate ( ) const

inline

Return the sampling frequency.

**Returns**

**OBAccelFullScaleRange** Return the sampling frequency.

Definition at line [295](#) of file [StreamProfile.hpp](#).

Referenced by [sampleRate\(\)](#).

◆ getIntrinsic()

**OBAccelIntrinsic** ob::AccelStreamProfile::getIntrinsic ( ) const

inline

get the intrinsic parameters of the stream.

### Returns

**OBAccelIntrinsic** Return the intrinsic parameters.

Definition at line **307** of file **StreamProfile.hpp**.

### ◆ fullScaleRange()

**OBAccelFullScaleRange** ob::AccelStreamProfile::fullScaleRange ( ) const

inline

Definition at line **316** of file **StreamProfile.hpp**.

Referenced by [getFullScaleRange\(\)](#).

### ◆ sampleRate()

**OBAccelSampleRate** ob::AccelStreamProfile::sampleRate ( ) const

inline

Definition at line **320** of file **StreamProfile.hpp**.

Referenced by [getSampleRate\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**StreamProfile.hpp**

# ob::Align Member List

This is the complete list of members for **ob::Align**, including all inherited members.

<b>Align</b> (OBStreamType alignToStreamType)	<b>ob::Align</b> inline
<b>as()</b>	<b>ob::Filter</b> inline
<b>callback_</b>	<b>ob::Filter</b> protected
<b>configSchemaVec_</b>	<b>ob::Filter</b> protected
<b>enable</b> (bool enable) const	<b>ob::Filter</b> inline virtual
<b>Filter()</b> =default	<b>ob::Filter</b> protected
<b>Filter</b> (ob_filter *impl)	<b>ob::Filter</b> inline explicit
<b>getAlignToStreamType()</b>	<b>ob::Align</b> inline
<b>getConfigSchema()</b> const	<b>ob::Filter</b> inline virtual
<b>getConfigSchemaVec()</b> const	<b>ob::Filter</b> inline virtual
<b>getConfigValue</b> (const std::string &configName) const	<b>ob::Filter</b> inline virtual
<b>getImpl()</b> const	<b>ob::Filter</b> inline
<b>getName()</b> const	<b>ob::Filter</b> inline virtual
<b>impl_</b>	<b>ob::Filter</b> protected
<b>init</b> (ob_filter *impl)	<b>ob::Filter</b> inline protected virtual
<b>is()</b>	<b>ob::Filter</b>
<b>isEnabled()</b> const	<b>ob::Filter</b> inline virtual
<b>name_</b>	<b>ob::Filter</b> protected
<b>process</b> (std::shared_ptr< const Frame > frame) const	<b>ob::Filter</b> inline virtual
<b>pushFrame</b> (std::shared_ptr< Frame > frame) const	<b>ob::Filter</b> inline virtual
<b>reset()</b> const	<b>ob::Filter</b> inline virtual
<b>setAlignToStreamProfile</b> (std::shared_ptr< const StreamProfile > profile)	<b>ob::Align</b> inline
<b>setCallBack</b> (FilterCallback callback)	<b>ob::Filter</b> inline virtual
<b>setConfigValue</b> (const std::string &configName, double value) const	<b>ob::Filter</b> inline virtual
<b>setMatchTargetResolution</b> (bool state)	<b>ob::Align</b> inline
<b>type()</b>	<b>ob::Filter</b> inline virtual
<b>~Align()</b> noexcept override=default	<b>ob::Align</b> virtual
<b>~Filter()</b> noexcept	<b>ob::Filter</b> inline virtual

# ob::Align Class Reference

**Align** for depth to other or other to depth. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::Align:

## Public Member Functions

**Align** (**OBStreamType** alignToStreamType)

virtual **~Align** () noexcept override=default

**OBStreamType** **getAlignToStreamType** ()

void **setMatchTargetResolution** (bool state)

Sets whether the output frame resolution should match the target resolution. When enabled, the output frame resolution will be adjusted to match (same as) the target resolution. When disabled, the output frame resolution will match the original resolution while maintaining the aspect ratio of the target resolution.

void **setAlignToStreamProfile** (std::shared\_ptr< const **StreamProfile** > profile)

Set the **Align** To Stream Profile.

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

**Align** for depth to other or other to depth.

Definition at line **413** of file **Filter.hpp**.

## Constructor & Destructor Documentation

- ◆ **Align()**

ob::Align::Align ( **OBStreamType** alignToStreamType )

inline

Definition at line **415** of file **Filter.hpp**.

◆ ~Align()

virtual ob::Align::~Align ( )

override virtual default noexcept

## Member Function Documentation

◆ getAlignToStreamType()

**OBStreamType** ob::Align::getAlignToStreamType ( )

inline

Definition at line **426** of file **Filter.hpp**.

Referenced by **getAlignToStreamType()**.

◆ setMatchTargetResolution()

void ob::Align::setMatchTargetResolution ( bool state )

inline

Sets whether the output frame resolution should match the target resolution. When enabled, the output frame resolution will be adjusted to match (same as) the target resolution. When disabled, the output frame resolution will match the original resolution while maintaining the aspect ratio of the target resolution.

### Parameters

**state** If true, output frame resolution will match the target resolution; otherwise, it will maintain the original resolution with the target's aspect ratio.

Definition at line **440** of file **Filter.hpp**.

◆ setAlignToStreamProfile()

```
void ob::Align::setAlignToStreamProfile ( std::shared_ptr<const StreamProfile> profile )
```

inline

Set the **Align** To Stream Profile.

It is useful when the align target stream dose not started (without any frame to get intrinsics and extrinsics).

### Parameters

**profile** The **Align** To Stream Profile.

Definition at line **450** of file **Filter.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::CameraParamList Member List

This is the complete list of members for **ob::CameraParamList**, including all inherited members.

<b>CameraParamList</b> (ob_camera_param_list_t *impl)	<b>ob::CameraParamList</b>	inline explicit
<b>count()</b>	<b>ob::CameraParamList</b>	inline
<b>getCameraParam</b> (uint32_t index)	<b>ob::CameraParamList</b>	inline
<b>getCount()</b>	<b>ob::CameraParamList</b>	inline
<b>~CameraParamList()</b> noexcept	<b>ob::CameraParamList</b>	inline

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::CameraParamList Class Reference

Class representing a list of camera parameters. [More...](#)

```
#include <Device.hpp>
```

## Public Member Functions

[\*\*CameraParamList\*\*](#) (ob\_camera\_param\_list\_t \*impl)

[\*\*~CameraParamList\*\*](#) () noexcept

uint32\_t [\*\*getCount\*\*](#) ()

Get the number of camera parameters in the list.

[\*\*OBCameraParam getCameraParam\*\*](#) (uint32\_t index)

Get the camera parameters for the specified index.

uint32\_t [\*\*count\*\*](#) ()

## Detailed Description

Class representing a list of camera parameters.

Definition at line [1421](#) of file [Device.hpp](#).

## Constructor & Destructor Documentation

◆ [\*\*CameraParamList\(\)\*\*](#)

ob::CameraParamList::CameraParamList ( ob\_camera\_param\_list\_t \* impl )

inline explicit

Definition at line [1426](#) of file [Device.hpp](#).

◆ [\*\*~CameraParamList\(\)\*\*](#)

ob::CameraParamList::~CameraParamList ( )

inline noexcept

Definition at line [1427](#) of file [Device.hpp](#).

# Member Function Documentation

## ◆ getCount()

`uint32_t ob::CameraParamList::getCount ( )`

inline

Get the number of camera parameters in the list.

### Returns

`uint32_t` the number of camera parameters in the list.

Definition at line [1438](#) of file [Device.hpp](#).

Referenced by [count\(\)](#).

## ◆ getCameraParam()

**OBCameraParam** `ob::CameraParamList::getCameraParam ( uint32_t index )`

inline

Get the camera parameters for the specified index.

### Parameters

`index` the index of the parameter group

### Returns

**OBCameraParam** the corresponding group parameters

Definition at line [1451](#) of file [Device.hpp](#).

## ◆ count()

`uint32_t ob::CameraParamList::count ( )`

inline

Definition at line [1460](#) of file [Device.hpp](#).

Referenced by [getCount\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Device.hpp](#)



## ob::ColorFrame Member List

This is the complete list of members for **ob::ColorFrame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>ColorFrame(const ob_frame *impl)</b>	<b>ob::ColorFrame</b>	inline explicit
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getHeight() const</b>	<b>ob::VideoFrame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getPixelAvailableBitSize() const</b>	<b>ob::VideoFrame</b>	inline
<b>getPixelType() const</b>	<b>ob::VideoFrame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getWidth() const</b>	<b>ob::VideoFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>height() const</b>	<b>ob::VideoFrame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	

<b>metadata()</b> const	<b>ob::Frame</b>	inline
<b>metadataSize()</b> const	<b>ob::Frame</b>	inline
<b>pixelAvailableBitSize()</b> const	<b>ob::VideoFrame</b>	inline
<b>systemTimeStamp()</b> const	<b>ob::Frame</b>	inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>VideoFrame</b> (const ob_frame *impl)	<b>ob::VideoFrame</b>	inline explicit
<b>width()</b> const	<b>ob::VideoFrame</b>	inline
<b>~ColorFrame()</b> noexcept override=default	<b>ob::ColorFrame</b>	
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~VideoFrame()</b> noexcept override=default	<b>ob::VideoFrame</b>	

# ob::ColorFrame Class Reference

Define the **ColorFrame** class, which inherits from the **VideoFrame** classd. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::ColorFrame:

## Public Member Functions

**ColorFrame** (const **ob\_frame** \*impl)

Construct a new **ColorFrame** object with a given pointer to the internal frame object.

**~ColorFrame** () noexcept override=default

Public Member Functions inherited from **ob::VideoFrame**

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **ColorFrame** class, which inherits from the **VideoFrame** classd.

Definition at line **481** of file **Frame.hpp**.

## Constructor & Destructor Documentation

- ◆ **ColorFrame()**

```
ob::ColorFrame::ColorFrame ( const ob_frame * impl )           inline explicit
```

Construct a new **ColorFrame** object with a given pointer to the internal frame object.

### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

Please use the **FrameFactory** to create a **Frame** object.

### Parameters

**impl** The pointer to the internal frame object.

Definition at line [493](#) of file **Frame.hpp**.

### ◆ ~ColorFrame()

```
ob::ColorFrame::~ColorFrame ( )           override default noexcept
```

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Frame.hpp**

## ob::ConfidenceFrame Member List

This is the complete list of members for **ob::ConfidenceFrame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>ConfidenceFrame(const ob_frame *impl)</b>	<b>ob::ConfidenceFrame</b>	inline explicit
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getHeight() const</b>	<b>ob::VideoFrame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getPixelAvailableBitSize() const</b>	<b>ob::VideoFrame</b>	inline
<b>getPixelType() const</b>	<b>ob::VideoFrame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getWidth() const</b>	<b>ob::VideoFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>height() const</b>	<b>ob::VideoFrame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	

<b>metadata()</b> const	<b>ob::Frame</b>	inline
<b>metadataSize()</b> const	<b>ob::Frame</b>	inline
<b>pixelAvailableBitSize()</b> const	<b>ob::VideoFrame</b>	inline
<b>systemTimeStamp()</b> const	<b>ob::Frame</b>	inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>VideoFrame</b> (const ob_frame *impl)	<b>ob::VideoFrame</b>	inline explicit
<b>width()</b> const	<b>ob::VideoFrame</b>	inline
<b>~ConfidenceFrame()</b> noexcept override=default	<b>ob::ConfidenceFrame</b>	
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~VideoFrame()</b> noexcept override=default	<b>ob::VideoFrame</b>	

# ob::ConfidenceFrame Class Reference

Define the **ConfidenceFrame** class, which inherits from the **VideoFrame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::ConfidenceFrame:

## Public Member Functions

**ConfidenceFrame** (const **ob\_frame** \*impl)

Construct a new **ConfidenceFrame** object with a given pointer to the internal frame object.

**~ConfidenceFrame** () noexcept override=default

Public Member Functions inherited from **ob::VideoFrame**

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **ConfidenceFrame** class, which inherits from the **VideoFrame** class.

Definition at line **560** of file **Frame.hpp**.

## Constructor & Destructor Documentation

- ◆ **ConfidenceFrame()**

```
ob::ConfidenceFrame::ConfidenceFrame ( const ob_frame * impl )  
    inline explicit
```

Construct a new **ConfidenceFrame** object with a given pointer to the internal frame object.

### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

Please use the **FrameFactory** to create a **Frame** object.

### Parameters

**impl** The pointer to the internal frame object.

Definition at line [573](#) of file **Frame.hpp**.

### ◆ ~ConfidenceFrame()

```
ob::ConfidenceFrame::~ConfidenceFrame ( )  
    override default noexcept
```

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Frame.hpp**

## ob::Config Member List

This is the complete list of members for **ob::Config**, including all inherited members.

**Config()**

**Config**(ob\_config\_t \*impl)

**disableAllStream()** const

**disableStream**(OBStreamType streamType) const

**disableStream**(OBSensorType sensorType) const

**enableAccelStream**(OBAccelFullScaleRange fullScaleRange=OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY, OBAccelFullScaleRange fullScaleRange=OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY, OBAccelFullScaleRange fullScaleRange=OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY)

**enableAllStream()**

**enableGyroStream**(OBGyroFullScaleRange fullScaleRange=OB\_GYRO\_FULL\_SCALE\_RANGE\_ANY, OBGyroFullScaleRange fullScaleRange=OB\_GYRO\_FULL\_SCALE\_RANGE\_ANY, OBGyroFullScaleRange fullScaleRange=OB\_GYRO\_FULL\_SCALE\_RANGE\_ANY)

**enableStream**(OBStreamType streamType) const

**enableStream**(OBSensorType sensorType) const

**enableStream**(std::shared\_ptr< const StreamProfile > streamProfile) const

**enableVideoStream**(OBStreamType streamType, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY)

**enableVideoStream**(OBSensorType sensorType, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY)

**getEnabledStreamProfileList()** const

**getImpl()** const

**setAlignMode**(OBAlignMode mode) const

**setDepthScaleRequire**(bool enable) const

**setFrameAggregateOutputMode**(OBFrameAggregateOutputMode mode) const

**~Config()** noexcept

# ob::Config Class Reference

**Config** class for configuring pipeline parameters. [More...](#)

```
#include <Pipeline.hpp>
```

## Public Member Functions

**Config ()**

Construct a new **Config** object.

**Config (ob\_config\_t \*impl)**

**~Config () noexcept**

Destroy the **Config** object.

**ob\_config\_t \* getImpl () const**

void **enableStream (OBStreamType streamType) const**

enable a stream with a specific stream type

void **enableStream (OBSensorType sensorType) const**

Enable a stream with a specific sensor type.

void **enableStream (std::shared\_ptr< const StreamProfile > streamProfile) const**

Enable a stream to be used in the pipeline.

void **enableVideoStream (OBStreamType streamType, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY, uint32\_t fps=OB\_FPS\_ANY, OBFormat format=OB\_FORMAT\_ANY) const**

Enable a video stream to be used in the pipeline.

void **enableVideoStream (OBSensorType sensorType, uint32\_t width=OB\_WIDTH\_ANY, uint32\_t height=OB\_HEIGHT\_ANY, uint32\_t fps=OB\_FPS\_ANY, OBFormat format=OB\_FORMAT\_ANY) const**

Enable a video stream to be used in the pipeline.

void **enableAccelStream (OBAccelFullScaleRange fullScaleRange=OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY, OBAccelSampleRate sampleRate=OB\_ACCEL\_SAMPLE\_RATE\_ANY) const**

Enable an accelerometer stream to be used in the pipeline.

```

void enableGyroStream (OBGyroFullScaleRange
    fullScaleRange=OB_GYRO_FULL_SCALE_RANGE_ANY,
    OBGyroSampleRate
    sampleRate=OB_GYRO_SAMPLE_RATE_ANY) const
    Enable a gyroscope stream to be used in the pipeline.

void enableAllStream ()
    Enable all streams to be used in the pipeline.

void disableStream (OBStreamType streamType) const
    Disable a stream to be used in the pipeline.

void disableStream (OBSensorType sensorType) const
    Disable a sensor stream to be used in the pipeline.

void disableAllStream () const
    Disable all streams to be used in the pipeline.

std::shared_ptr<StreamProfileList> getEnabledStreamProfileList () const
    Get the Enabled Stream Profile List.

void setAlignMode (OBAlignMode mode) const
    Set the alignment mode.

void setDepthScaleRequire (bool enable) const
    Set whether the depth needs to be scaled after setting D2C.

void  setFrameAggregateOutputMode
    (OBFrameAggregateOutputMode mode) const
    Set the frame aggregation output mode for the pipeline
    configuration.

```

## Detailed Description

**Config** class for configuring pipeline parameters.

The **Config** class provides an interface for configuring pipeline parameters.

Definition at line **28** of file **Pipeline.hpp**.

## Constructor & Destructor Documentation

◆ **Config()** [1/2]

ob::Config::Config ( ) inline

Construct a new **Config** object.

Definition at line **36** of file [Pipeline.hpp](#).

◆ **Config()** [2/2]

ob::Config::Config ( ob\_config\_t \* impl ) inline explicit

Definition at line **42** of file [Pipeline.hpp](#).

◆ **~Config()**

ob::Config::~Config ( ) inline noexcept

Destroy the **Config** object.

Definition at line **47** of file [Pipeline.hpp](#).

## Member Function Documentation

◆ **getImpl()**

ob\_config\_t \* ob::Config::getImpl ( ) const inline

Definition at line **53** of file [Pipeline.hpp](#).

◆ **enableStream()** [1/3]

```
void ob::Config::enableStream( OBStreamType streamType ) const inline
```

enable a stream with a specific stream type

#### Parameters

**streamType** The stream type to be enabled

Definition at line **62** of file [Pipeline.hpp](#).

Referenced by [enableStream\(\)](#).

#### ◆ [enableStream\(\)](#) [2/3]

```
void ob::Config::enableStream( OBSensorType sensorType ) const inline
```

Enable a stream with a specific sensor type.

Will convert sensor type to stream type automatically.

#### Parameters

**sensorType** The sensor type to be enabled

Definition at line **74** of file [Pipeline.hpp](#).

#### ◆ [enableStream\(\)](#) [3/3]

```
void ob::Config::enableStream( std::shared_ptr<const StreamProfile> streamProfile ) const inline
```

Enable a stream to be used in the pipeline.

#### Parameters

**streamProfile** The stream configuration to be enabled

Definition at line **84** of file [Pipeline.hpp](#).

#### ◆ [enableVideoStream\(\)](#) [1/2]

```
void ob::Config::enableVideoStream( OBStreamType streamType,  
                                  uint32_t      width = OB_WIDTH_ANY,  
                                  uint32_t      height = OB_HEIGHT_ANY,  
                                  uint32_t      fps = OB_FPS_ANY,  
                                  OBFormat     format = OB_FORMAT_ANY) const  
{  
    // Implementation details...  
}
```

inline

Enable a video stream to be used in the pipeline.

This function allows users to enable a video stream with customizable parameters. If no parameters are specified, the stream will be enabled with default resolution settings. Users who wish to set custom resolutions should refer to the product manual, as available resolutions vary by camera model.

## Parameters

- streamType** The video stream type.
- width** The video stream width (default is OB\_WIDTH\_ANY, which selects the default resolution).
- height** The video stream height (default is OB\_HEIGHT\_ANY, which selects the default resolution).
- fps** The video stream frame rate (default is OB\_FPS\_ANY, which selects the default frame rate).
- format** The video stream format (default is OB\_FORMAT\_ANY, which selects the default format).

Definition at line 104 of file [Pipeline.hpp](#).

Referenced by [enableVideoStream\(\)](#).

◆ [enableVideoStream\(\)](#) [2/2]

```

void ob::Config::enableVideoStream( OBSensorType sensorType,
                                    uint32_t      width = OB_WIDTH_ANY,
                                    uint32_t      height = OB_HEIGHT_ANY,
                                    uint32_t      fps = OB_FPS_ANY,
                                    OBFormat     format = OB_FORMAT_ANY) const
                                            inline

```

Enable a video stream to be used in the pipeline.

Will convert sensor type to stream type automatically.

## Parameters

- sensorType** The sensor type to be enabled.
- width** The video stream width (default is OB\_WIDTH\_ANY, which selects the default resolution).
- height** The video stream height (default is OB\_HEIGHT\_ANY, which selects the default resolution).
- fps** The video stream frame rate (default is OB\_FPS\_ANY, which selects the default frame rate).
- format** The video stream format (default is OB\_FORMAT\_ANY, which selects the default format).

Definition at line 121 of file [Pipeline.hpp](#).

## ◆ enableAccelStream()

```

void
ob::Config::enableAccelStream( OBAccelFullScaleRange fullScaleRange = OB_ACCEL_FULL_SCALE_RANGE_A
                                OBAccelSampleRate      sampleRate = OB_ACCEL_SAMPLE_RATE_ANY) const

```

Enable an accelerometer stream to be used in the pipeline.

This function allows users to enable an accelerometer stream with customizable parameters. If no parameters are specified, the stream will be enabled with default settings. Users who wish to set custom full-scale ranges or sample rates should refer to the product manual, as available settings vary by device model.

## Parameters

- fullScaleRange** The full-scale range of the accelerometer (default is OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY, which selects the default range).
- sampleRate** The sample rate of the accelerometer (default is OB\_ACCEL\_SAMPLE\_RATE\_ANY, which selects the default rate).

Definition at line 137 of file [Pipeline.hpp](#).

## ◆ enableGyroStream()

```
void  
ob::Config::enableGyroStream ( OBGyroFullScaleRange fullScaleRange = OB_GYRO_FULL_SCALE_RANGE_ANY  
                                OBGyroSampleRate      sampleRate = OB_GYRO_SAMPLE_RATE_ANY ) const
```

Enable a gyroscope stream to be used in the pipeline.

This function allows users to enable a gyroscope stream with customizable parameters. If no parameters are specified, the stream will be enabled with default settings. Users who wish to set custom full-scale ranges or sample rates should refer to the product manual, as available settings vary by device model.

### Parameters

**fullScaleRange** The full-scale range of the gyroscope (default is OB\_GYRO\_FULL\_SCALE\_RANGE\_ANY, which selects the default range).

**sampleRate** The sample rate of the gyroscope (default is OB\_GYRO\_SAMPLE\_RATE\_ANY, which selects the default rate).

Definition at line [154](#) of file [Pipeline.hpp](#).

## ◆ enableAllStream()

```
void ob::Config::enableAllStream ( )
```

inline

Enable all streams to be used in the pipeline.

### Deprecated

Use `enableStream(std::shared_ptr<StreamProfile> streamProfile)` instead

Definition at line [164](#) of file [Pipeline.hpp](#).

## ◆ disableStream() [1/2]

```
void ob::Config::disableStream( OBStreamType streamType ) const inline
```

Disable a stream to be used in the pipeline.

#### Parameters

**streamType** The stream configuration to be disabled

Definition at line **175** of file [Pipeline.hpp](#).

Referenced by [disableStream\(\)](#).

#### ◆ [disableStream\(\)](#) [2/2]

```
void ob::Config::disableStream( OBSensorType sensorType ) const inline
```

Disable a sensor stream to be used in the pipeline.

Will convert sensor type to stream type automatically.

#### Parameters

**sensorType** The sensor configuration to be disabled

Definition at line **187** of file [Pipeline.hpp](#).

#### ◆ [disableAllStream\(\)](#)

```
void ob::Config::disableAllStream( ) const inline
```

Disable all streams to be used in the pipeline.

Definition at line **195** of file [Pipeline.hpp](#).

#### ◆ [getEnabledStreamProfileList\(\)](#)

```
std::shared_ptr<StreamProfileList> ob::Config::getEnabledStreamProfileList( ) const inline
```

Get the Enabled Stream Profile List.

#### Returns

```
std::shared_ptr<StreamProfileList>
```

Definition at line **206** of file [Pipeline.hpp](#).

#### ◆ **setAlignMode()**

```
void ob::Config::setAlignMode( OBAlignMode mode ) const inline
```

Set the alignment mode.

#### Parameters

**mode** The alignment mode

Definition at line **218** of file [Pipeline.hpp](#).

#### ◆ **setDepthScaleRequire()**

```
void ob::Config::setDepthScaleRequire( bool enable ) const inline
```

Set whether the depth needs to be scaled after setting D2C.

#### Parameters

**enable** Whether scaling is required

Definition at line **229** of file [Pipeline.hpp](#).

#### ◆ **setFrameAggregateOutputMode()**

```
void ob::Config::setFrameAggregateOutputMode ( OBFrameAggregateOutputMode mode ) const inline
```

Set the frame aggregation output mode for the pipeline configuration.

The processing strategy when the **FrameSet** generated by the frame aggregation function does not contain the frames of all opened streams (which can be caused by different frame rates of each stream, or by the loss of frames of one stream): drop directly or output to the user.

## Parameters

**mode** The frame aggregation output mode to be set (default mode is  
**OB\_FRAME\_AGGREGATE\_OUTPUT\_ANY\_SITUATION**)

Definition at line [242](#) of file [Pipeline.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Pipeline.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::Context Member List

This is the complete list of members for **ob::Context**, including all inherited members.

<b>Context</b> (const char *configPath="")	<b>ob::Context</b>	inline explicit
<b>createNetDevice</b> (const char *address, uint16_t port) const	<b>ob::Context</b>	inline
<b>DeviceChangedCallback</b> typedef	<b>ob::Context</b>	
<b>enableDeviceClockSync</b> (uint64_t repeatIntervalMsec) const	<b>ob::Context</b>	inline
<b>enableNetDeviceEnumeration</b> (bool enable) const	<b>ob::Context</b>	inline
<b>freeIdleMemory</b> () const	<b>ob::Context</b>	inline
<b>LogCallback</b> typedef	<b>ob::Context</b>	
<b>queryDeviceList</b> () const	<b>ob::Context</b>	inline
<b>setDeviceChangedCallback</b> (DeviceChangedCallback callback)	<b>ob::Context</b>	inline
<b>setExtensionsDirectory</b> (const char *directory)	<b>ob::Context</b>	inline static
<b>setLoggerSeverity</b> (OBLogSeverity severity)	<b>ob::Context</b>	inline static
<b>setLoggerToCallback</b> (OBLogSeverity severity, LogCallback callback)	<b>ob::Context</b>	inline static
<b>setLoggerToConsole</b> (OBLogSeverity severity)	<b>ob::Context</b>	inline static
<b>setLoggerToFile</b> (OBLogSeverity severity, const char *directory)	<b>ob::Context</b>	inline static
<b>setUvcBackendType</b> (OBUvcBackendType type) const	<b>ob::Context</b>	inline
<b>~Context</b> () noexcept	<b>ob::Context</b>	inline

# ob::Context Class Reference

```
#include <Context.hpp>
```

## Public Types

```
typedef std::function< void(std::shared_ptr< DeviceList > removedList, std::shared_ptr< DeviceList > addedL
```

```
typedef std::function< void(OBLogSeverity severity, const char *logM
```

## Public Member Functions

```
Context (const char *configPath="")
```

Context constructor.

```
~Context () noexcept
```

Context destructor.

```
std::shared_ptr< DeviceList > queryDeviceList () const
```

Queries the enumerated device list.

```
void enableNetDeviceEnumeration (bool enable) const
```

enable or disable net device enumeration.

```
std::shared_ptr< Device > createNetDevice (const char *address, uint16_t port) const
```

Creates a network device with the specified IP address and port.

```
void setDeviceChangedCallback (DeviceChangedCallback callback)
```

Set the device plug-in callback function.

```
void enableDeviceClockSync (uint64_t repeatIntervalMsec) const
```

Activates device clock synchronization to synchronize the clock of the host and all created devices (if supported).

```
void freeIdleMemory () const
```

Frees idle memory from the internal frame memory pool.

```
void setUvcBackendType (OBUvcBackendType type) const
```

For linux, there are two ways to enable the UVC backend: libuvc and v4l2.

This function is used to set the backend type.

## Static Public Member Functions

static void **setLoggerSeverity** (**OBLogSeverity** severity)

Set the level of the global log, which affects both the log level output to the console, output to the file and output the user defined callback.

static void **setLoggerToFile** (**OBLogSeverity** severity, const char \*directory)

Set log output to a file.

static void **setLoggerToConsole** (**OBLogSeverity** severity)

Set log output to the console.

static void **setLoggerToCallback** (**OBLogSeverity** severity, **LogCallback** callback)

Set the logger to callback.

static void **setExtensionsDirectory** (const char \*directory)

Set the extensions directory.

## Detailed Description

Definition at line **26** of file **Context.hpp**.

## Member Typedef Documentation

### ◆ DeviceChangedCallback

```
typedef std::function<void(std::shared_ptr<DeviceList> removedList, std::shared_ptr<DeviceList> addedList)>
ob::Context::DeviceChangedCallback
```

Type definition for the device changed callback function.

### Parameters

**removedList** The list of removed devices.

**addedList** The list of added devices.

Definition at line **34** of file **Context.hpp**.

### ◆ LogCallback

```
typedef std::function<void(OBLogSeverity severity, const char *logMsg)> ob::Context::LogCallback
```

Type definition for the log output callback function.

#### Parameters

**severity** The current callback log level.

**logMsg** The log message.

Definition at line [42](#) of file [Context.hpp](#).

## Constructor & Destructor Documentation

#### ◆ Context()

```
ob::Context::Context ( const char * configPath = "" )inline explicit
```

**Context** constructor.

The **Context** class is a management class that describes the runtime of the SDK. It is responsible for applying and releasing resources for the SDK. The context has the ability to manage multiple devices, enumerate devices, monitor device callbacks, and enable functions such as multi-device synchronization.

#### Parameters

[in] **configPath** The path to the configuration file. If the path is empty, the default configuration will be used.

Definition at line [58](#) of file [Context.hpp](#).

#### ◆ ~Context()

```
ob::Context::~Context ( )inline noexcept
```

**Context** destructor.

Definition at line [67](#) of file [Context.hpp](#).

## Member Function Documentation

#### ◆ queryDeviceList()

```
std::shared_ptr<DeviceList> ob::Context::queryDeviceList ( ) const inline
```

Queries the enumerated device list.

### Returns

```
std::shared_ptr<DeviceList> A pointer to the device list class.
```

Definition at line **78** of file [Context.hpp](#).

### ◆ enableNetDeviceEnumeration()

```
void ob::Context::enableNetDeviceEnumeration ( bool enable ) const inline
```

enable or disable net device enumeration.

after enable, the net device will be discovered automatically and can be retrieved by [queryDeviceList](#). The default state can be set in the configuration file.

### Attention

Net device enumeration by gvcp protocol, if the device is not in the same subnet as the host, it will be discovered but cannot be connected.

### Parameters

[in] **enable** true to enable, false to disable

Definition at line **94** of file [Context.hpp](#).

### ◆ createNetDevice()

```
std::shared_ptr<Device> ob::Context::createNetDevice ( const char * address,  
                                              uint16_t      port ) const  
                                                inline
```

Creates a network device with the specified IP address and port.

### Parameters

- [in] **address** The IP address, ipv4 only. such as "192.168.1.10"
- [in] **port** The port number, currently only support 8090

### Returns

std::shared\_ptr<Device> The created device object.

Definition at line [107](#) of file [Context.hpp](#).

### ◆ setDeviceChangedCallback()

```
void ob::Context::setDeviceChangedCallback ( DeviceChangedCallback callback )  
                                                inline
```

Set the device plug-in callback function.

### Attention

This function supports multiple callbacks. Each call to this function adds a new callback to an internal list.

### Parameters

- callback** The function triggered when the device is plugged and unplugged.

Definition at line [120](#) of file [Context.hpp](#).

### ◆ enableDeviceClockSync()

```
void ob::Context::enableDeviceClockSync ( uint64_t repeatIntervalMsec ) const  
                                                inline
```

Activates device clock synchronization to synchronize the clock of the host and all created devices (if supported).

### Parameters

- repeatIntervalMsec** The interval for auto-repeated synchronization, in milliseconds. If the value is 0, synchronization is performed only once.

Definition at line [132](#) of file [Context.hpp](#).

◆ **freeIdleMemory()**

void ob::Context::freeIdleMemory ( ) const

inline

Frees idle memory from the internal frame memory pool.

The SDK includes an internal frame memory pool that caches memory allocated for frames received from devices.

Definition at line [142](#) of file [Context.hpp](#).

◆ **setUvcBackendType()**

void ob::Context::setUvcBackendType ( **OBUvcBackendType** type ) const

inline

For linux, there are two ways to enable the UVC backend: libuvc and v4l2. This function is used to set the backend type.

It is effective when the new device is created.

**Attention**

This interface is only available for Linux.

**Parameters**

[in] **type** The backend type to be used.

Definition at line [156](#) of file [Context.hpp](#).

◆ **setLoggerSeverity()**

void ob::Context::setLoggerSeverity ( **OBLogSeverity** severity )

inline static

Set the level of the global log, which affects both the log level output to the console, output to the file and output the user defined callback.

**Parameters**

**severity** The log output level.

Definition at line [168](#) of file [Context.hpp](#).

◆ **setLoggerToFile()**

```
void ob::Context::setLoggerToFile ( OBLogSeverity severity,  
                                    const char *      directory )  
                                         inline static
```

Set log output to a file.

#### Parameters

**severity** The log level output to the file.

**directory** The log file output path. If the path is empty, the existing settings will continue to be used (if the existing configuration is also empty, the log will not be output to the file).

Definition at line [181](#) of file [Context.hpp](#).

#### ◆ setLoggerToConsole()

```
void ob::Context::setLoggerToConsole ( OBLogSeverity severity )  
                                         inline static
```

Set log output to the console.

#### Parameters

**severity** The log level output to the console.

Definition at line [192](#) of file [Context.hpp](#).

#### ◆ setLoggerToCallback()

```
void ob::Context::setLoggerToCallback ( OBLogSeverity severity,  
                                       LogCallback      callback )  
                                         inline static
```

Set the logger to callback.

#### Parameters

**severity** The callback log level.

**callback** The callback function.

Definition at line [204](#) of file [Context.hpp](#).

#### ◆ setExtensionsDirectory()

```
void ob::Context::setExtensionsDirectory ( const char * directory )
```

inline static

Set the extensions directory.

The extensions directory is used to search for dynamic libraries that provide additional functionality to the SDK, such as the **Frame** filters.

### Attention

Should be called before creating the context and pipeline, otherwise the default extensions directory (./extensions) will be used.

### Parameters

**directory** Path to the extensions directory. If the path is empty, the existing settings will continue to be used (if the existing

Definition at line **219** of file **Context.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Context.hpp**

## ob::CoordinateTransformHelper Member List

This is the complete list of members for **ob::CoordinateTransformHelper**, including all inherited members.

**calibration2dTo2d**(const OBCalibrationParam calibrationParam, const OBPoint2f sourcePoint2f, const float sou  
**calibration2dTo3d**(const OBCalibrationParam calibrationParam, const OBPoint2f sourcePoint2f, const float sou  
**calibration3dTo2d**(const OBCalibrationParam calibrationParam, const OBPoint3f sourcePoint3f, const OBSensorType sensorType, const OBCameraIntrinsic cameraIntrinsic, const OBCameraExtrinsic cameraExtrinsic, const float depthValue, const OBPoint2f targetPoint2f)  
**calibration3dTo3d**(const OBCalibrationParam calibrationParam, const OBPoint3f sourcePoint3f, const OBSensorType sensorType, const OBCameraIntrinsic cameraIntrinsic, const OBCameraExtrinsic cameraExtrinsic, const float depthValue, const OBPoint3f targetPoint3f)  
**transformation2dto2d**(const OBPoint2f source\_point2f, const float source\_depth\_pixel\_value, const OBCamer  
**transformation2dto3d**(const OBPoint2f source\_point2f, const float source\_depth\_pixel\_value, const OBCamer  
**transformation3dto2d**(const OBPoint3f source\_point3f, const OBCameraIntrinsic target\_intrinsic, const OBCamer  
**transformation3dto3d**(const OBPoint3f source\_point3f, OBExtrinsic extrinsic, OBPoint3f \*target\_point3f)  
**transformationDepthFrameToColorCamera**(std::shared\_ptr<ob::Device> device, std::shared\_ptr<ob::Frame> frame, const void \*depthImageData, void \*pointCloudData, const void \*colorImage)  
**transformationDepthToPointCloud**(OBXYTables \*xyTables, const void \*depthImageData, void \*PointCloudData, const void \*colorImage)  
**transformationDepthToRGBDPointCloud**(OBXYTables \*xyTables, const void \*depthImageData, const void \*colorImage, void \*pointCloudData, void \*colorImage)  
**transformationInitXYTables**(const OBCalibrationParam calibrationParam, const OBSensorType sensorType, const OBCameraIntrinsic cameraIntrinsic, const OBCameraExtrinsic cameraExtrinsic)

## ob::CoordinateTransformHelper Class Reference

```
#include <Utils.hpp>
```

## Static Public Member Functions

static bool **transformation3dto3d** (const **OBPoint3f** source\_point3f, **OBExtrinsic** extrinsic, **OBPoint3f** \*target\_point3f)  
Transform a 3d point of a source coordinate system into a 3d point of the target coordinate system.

static bool **transformation2dto3d** (const **OBPoint2f** source\_point2f, const float source\_depth\_pixel\_value, const **OBCameraIntrinsic** source\_intrinsic, **OBExtrinsic** extrinsic, **OBPoint3f** \*target\_point3f)  
Transform a 2d pixel coordinate with an associated depth value of the source camera into a 3d point of the target coordinate system.

static bool **transformation3dto2d** (const **OBPoint3f** source\_point3f, const **OBCameraIntrinsic** target\_intrinsic, const **OBCameraDistortion** target\_distortion, **OBExtrinsic** extrinsic, **OBPoint2f** \*target\_point2f)  
Transform a 3d point of a source coordinate system into a 2d pixel coordinate of the target camera.

static bool **transformation2dto2d** (const **OBPoint2f** source\_point2f, const float source\_depth\_pixel\_value, const **OBCameraIntrinsic** source\_intrinsic, const **OBCameraDistortion** source\_distortion, const **OBCameraIntrinsic** target\_intrinsic, const **OBCameraDistortion** target\_distortion, **OBExtrinsic** extrinsic, **OBPoint2f** \*target\_point2f)  
Transform a 2d pixel coordinate with an associated depth value of the source camera into a 2d pixel coordinate of the target camera.

static bool **calibration3dTo3d** (const **OBCalibrationParam** calibrationParam, const **OBPoint3f** sourcePoint3f, const **OBSensorType** sourceSensorType, const **OBSensorType** targetSensorType, **OBPoint3f** \*targetPoint3f)

static bool **calibration2dTo3d** (const **OBCalibrationParam** calibrationParam, const **OBPoint2f** sourcePoint2f, const float sourceDepthPixelValue, const **OBSensorType** sourceSensorType, const **OBSensorType** targetSensorType, **OBPoint3f** \*targetPoint3f)

static bool **calibration3dTo2d** (const **OBCalibrationParam** calibrationParam, const **OBPoint3f** sourcePoint3f, const **OBSensorType** sourceSensorType, const **OBSensorType** targetSensorType, **OBPoint2f** \*targetPoint2f)

```

static bool calibration2dTo2d (const OBCalibrationParam calibrationParam,
                           const OBPoint2f sourcePoint2f, const float sourceDepthPixelValue,
                           const OBSensorType sourceSensorType, const OBSensorType
                           targetSensorType, OBPoint2f *targetPoint2f)

static std::shared_ptr< ob::Frame > transformationDepthFrameToColorCamera (std::shared_ptr<
                           ob::Device > device, std::shared_ptr< ob::Frame > depthFrame,
                           uint32_t targetColorCameraWidth, uint32_t
                           targetColorCameraHeight)

static bool transformationInitXYTables (const OBCalibrationParam
                                       calibrationParam, const OBSensorType sensorType, float *data,
                                       uint32_t *dataSize, OBXYTables *xyTables)

static void transformationDepthToPointCloud (OBXYTables *xyTables,
                                              const void *depthImageData, void *pointCloudData)

static void transformationDepthToRGBDPointCloud (OBXYTables
                                              *xyTables, const void *depthImageData, const void
                                              *colorImageData, void *pointCloudData)

```

## Detailed Description

Definition at line **20** of file **Utils.hpp**.

## Member Function Documentation

### ◆ transformation3dto3d()

```

bool ob::CoordinateTransformHelper::transformation3dto3d ( const OBPoint3f source_point3f,
                                                       OBExtrinsic      extrinsic,
                                                       OBPoint3f *       target_point3f )           inline static

```

Transform a 3d point of a source coordinate system into a 3d point of the target coordinate system.

#### Parameters

- [in] **source\_point3f** Source 3d point value
- [in] **extrinsic** Transformation matrix from source to target
- [out] **target\_point3f** Target 3d point value

#### Returns

bool Transform result

Definition at line **31** of file **Utils.hpp**.

◆ transformation2dto3d()

```
bool  
ob::CoordinateTransformHelper::transformation2dto3d ( const OBPoint2f source_point2f,  
                                                    const float source_depth_pixel_value,  
                                                    const OBCameraIntrinsic source_intrinsic,  
                                                    OBExtrinsic extrinsic,  
                                                    OBPoint3f* target_point3f ) inline
```

Transform a 2d pixel coordinate with an associated depth value of the source camera into a 3d point of the target coordinate system.

#### Parameters

[in] <b>source_intrinsic</b>	Source intrinsic parameters
[in] <b>source_point2f</b>	Source 2d point value
[in] <b>source_depth_pixel_value</b>	The depth of sourcePoint2f in millimeters
[in] <b>extrinsic</b>	Transformation matrix from source to target
[out] <b>target_point3f</b>	Target 3d point value

#### Returns

bool Transform result

Definition at line 49 of file [Utils.hpp](#).

◆ transformation3dto2d()

```
bool  
ob::CoordinateTransformHelper::transformation3dto2d ( const OBPoint3f source_point3f,  
                                                 const OBCameraIntrinsic target_intrinsic,  
                                                 const OBCameraDistortion target_distortion,  
                                                 OBExtrinsic extrinsic,  
                                                 OBPoint2f * target_point2f ) inline static
```

Transform a 3d point of a source coordinate system into a 2d pixel coordinate of the target camera.

## Parameters

- [in] **source\_point3f** Source 3d point value
- [in] **target\_intrinsic** Target intrinsic parameters
- [in] **target\_distortion** Target distortion parameters
- [in] **extrinsic** Transformation matrix from source to target
- [out] **target\_point2f** Target 2d point value

## Returns

bool Transform result

Definition at line **68** of file [Utils.hpp](#).

◆ [transformation2dto2d\(\)](#)

```

bool
ob::CoordinateTransformHelper::transformation2dto2d ( const OBPoint2f source_point2f,
                                                    const float source_depth_pixel_value,
                                                    const OBCameraIntrinsic source_intrinsic,
                                                    const OBCameraDistortion source_distortion,
                                                    const OBCameraIntrinsic target_intrinsic,
                                                    const OBCameraDistortion target_distortion,
                                                    OBExtrinsic extrinsic,
OBPoint2f* target_point2f ) inli

```

Transform a 2d pixel coordinate with an associated depth value of the source camera into a 2d pixel coordinate the target camera.

### Parameters

[in] <b>source_intrinsic</b>	Source intrinsic parameters
[in] <b>source_distortion</b>	Source distortion parameters
[in] <b>source_point2f</b>	Source 2d point value
[in] <b>source_depth_pixel_value</b>	The depth of sourcePoint2f in millimeters
[in] <b>target_intrinsic</b>	Target intrinsic parameters
[in] <b>target_distortion</b>	Target distortion parameters
[in] <b>extrinsic</b>	Transformation matrix from source to target
[out] <b>target_point2f</b>	Target 2d point value

### Returns

bool Transform result

Definition at line **90** of file [Utils.hpp](#).

### ◆ calibration3dTo3d()

```

bool
ob::CoordinateTransformHelper::calibration3dTo3d ( const OBCalibrationParam calibrationParam,
                                                 const OBPoint3f sourcePoint3f,
                                                 const OBSensorType sourceSensorType,
                                                 const OBSensorType targetSensorType,
OBPoint3f* targetPoint3f ) inline static

```

Definition at line **102** of file [Utils.hpp](#).

◆ calibration2dTo3d()

```
bool  
ob::CoordinateTransformHelper::calibration2dTo3d ( const OBCalibrationParam calibrationParam,  
                                                 const OBPoint2f sourcePoint2f,  
                                                 const float sourceDepthPixelValue,  
                                                 const OBSensorType sourceSensorType,  
                                                 const OBSensorType targetSensorType,  
                                                 OBPoint3f* targetPoint3f )           inline static
```

Definition at line 110 of file [Utils.hpp](#).

◆ calibration3dTo2d()

```
bool  
ob::CoordinateTransformHelper::calibration3dTo2d ( const OBCalibrationParam calibrationParam,  
                                                 const OBPoint3f sourcePoint3f,  
                                                 const OBSensorType sourceSensorType,  
                                                 const OBSensorType targetSensorType,  
                                                 OBPoint2f* targetPoint2f )           inline static
```

Definition at line 119 of file [Utils.hpp](#).

◆ calibration2dTo2d()

```
bool  
ob::CoordinateTransformHelper::calibration2dTo2d ( const OBCalibrationParam calibrationParam,  
                                                 const OBPoint2f sourcePoint2f,  
                                                 const float sourceDepthPixelValue,  
                                                 const OBSensorType sourceSensorType,  
                                                 const OBSensorType targetSensorType,  
                                                 OBPoint2f* targetPoint2f )           inline static
```

Definition at line 127 of file [Utils.hpp](#).

◆ transformationDepthFrameToColorCamera()

```
std::shared_ptr<ob::Frame>
ob::CoordinateTransformHelper::transformationDepthFrameToColorCamera ( std::shared_ptr<ob::Device> device,
                                                                     std::shared_ptr<ob::Frame> depthFrame,
                                                                     uint32_t targetColor,
                                                                     uint32_t targetColor )
```

Definition at line [136](#) of file [Utils.hpp](#).

#### ◆ transformationInitXYTables()

```
bool
ob::CoordinateTransformHelper::transformationInitXYTables ( const OBCalibrationParam calibrationParam,
                                                               const OBSensorType sensorType,
                                                               float * data,
                                                               uint32_t * dataSize,
                                                               OBXYTables * xyTables ) inline
```

Definition at line [148](#) of file [Utils.hpp](#).

#### ◆ transformationDepthToPointCloud()

```
void
ob::CoordinateTransformHelper::transformationDepthToPointCloud ( OBXYTables * xyTables,
                                                               const void * depthImageData,
                                                               void * pointCloudData ) inline static
```

Definition at line [156](#) of file [Utils.hpp](#).

#### ◆ transformationDepthToRGBOBDPointCloud()

```
void
ob::CoordinateTransformHelper::transformationDepthToRGBDPointCloud ( OBXYTables * xyTables,
                                                               const void * depthImageData,
                                                               const void * colorImageData,
                                                               void * pointCloudData )

```

inline

Definition at line **162** of file [Utils.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Utils.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::DecimationFilter Member List

This is the complete list of members for **ob::DecimationFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>DecimationFilter()</b>	<b>ob::DecimationFilter</b>	inline
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>getScaleRange()</b>	<b>ob::DecimationFilter</b>	inline
<b>getScaleValue()</b>	<b>ob::DecimationFilter</b>	inline
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virtu
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt; const Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt; Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setScaleValue(uint8_t value)</b>	<b>ob::DecimationFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~DecimationFilter() noexcept override=default</b>	<b>ob::DecimationFilter</b>	virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual

# ob::DecimationFilter Class Reference

Decimation filter, reducing complexity by subsampling depth maps and losing depth details. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::DecimationFilter:

## Public Member Functions

[\*\*DecimationFilter\(\)\*\*](#)

virtual [\*\*~DecimationFilter\(\)\*\*](#) noexcept override=default

void [\*\*setScaleValue\*\*](#) (uint8\_t value)

Set the decimation filter scale value.

uint8\_t [\*\*getScaleValue\*\*](#) ()

Get the decimation filter scale value.

[\*\*OBUInt8PropertyRange getScaleRange\(\)\*\*](#)

Get the property range of the decimation filter scale value.

Public Member Functions inherited from [ob::Filter](#)

## Additional Inherited Members

Protected Member Functions inherited from [ob::Filter](#)

Protected Attributes inherited from [ob::Filter](#)

## Detailed Description

Decimation filter, reducing complexity by subsampling depth maps and losing depth details.

Definition at line [569](#) of file [Filter.hpp](#).

## Constructor & Destructor Documentation

◆ [\*\*DecimationFilter\(\)\*\*](#)

ob::DecimationFilter::DecimationFilter( )

inline

Definition at line [571](#) of file [Filter.hpp](#).

◆ ~DecimationFilter()

virtual ob::DecimationFilter::~DecimationFilter( )

override virtual default noexcept

## Member Function Documentation

◆ setScaleValue()

void ob::DecimationFilter::setScaleValue ( uint8\_t value )

inline

Set the decimation filter scale value.

### Parameters

**value** The decimation filter scale value.

Definition at line [585](#) of file [Filter.hpp](#).

Referenced by [setScaleValue\(\)](#).

◆ getScaleValue()

uint8\_t ob::DecimationFilter::getScaleValue ( )

inline

Get the decimation filter scale value.

Definition at line [592](#) of file [Filter.hpp](#).

◆ getScaleRange()

**OB UInt8PropertyRange** ob::DecimationFilter::getScaleRange ( )

inline

Get the property range of the decimation filter scale value.

Definition at line [599](#) of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)



## ob::DepthFrame Member List

This is the complete list of members for **ob::DepthFrame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>DepthFrame(const ob_frame *impl)</b>	<b>ob::DepthFrame</b>	inline explicit
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getHeight() const</b>	<b>ob::VideoFrame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getPixelAvailableBitSize() const</b>	<b>ob::VideoFrame</b>	inline
<b>getPixelType() const</b>	<b>ob::VideoFrame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getValueScale() const</b>	<b>ob::DepthFrame</b>	inline
<b>getWidth() const</b>	<b>ob::VideoFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>height() const</b>	<b>ob::VideoFrame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual

<b>is()</b> const	<b>ob::Frame</b>
<b>metadata()</b> const	<b>ob::Frame</b> inline
<b>metadataSize()</b> const	<b>ob::Frame</b> inline
<b>pixelAvailableBitSize()</b> const	<b>ob::VideoFrame</b> inline
<b>systemTimeStamp()</b> const	<b>ob::Frame</b> inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b> inline
<b>timeStamp()</b> const	<b>ob::Frame</b> inline
<b>timeStampUs()</b> const	<b>ob::Frame</b> inline
<b>type()</b> const	<b>ob::Frame</b> inline
<b>VideoFrame</b> (const ob_frame *impl)	<b>ob::VideoFrame</b> inline explicit
<b>width()</b> const	<b>ob::VideoFrame</b> inline
<b>~DepthFrame()</b> noexcept override=default	<b>ob::DepthFrame</b>
<b>~Frame()</b> noexcept	<b>ob::Frame</b> inline virtual
<b>~VideoFrame()</b> noexcept override=default	<b>ob::VideoFrame</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::DepthFrame Class Reference

Define the **DepthFrame** class, which inherits from the **VideoFrame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::DepthFrame:

## Public Member Functions

**DepthFrame** (const **ob\_frame** \*impl)

Construct a new **DepthFrame** object with a given pointer to the internal frame object.

**~DepthFrame** () noexcept override=default

float **getValueScale** () const

Get the value scale of the depth frame. The pixel value of depth frame is multiplied by the scale to give a depth value in millimeters. For example, if valueScale=0.1 and a certain coordinate pixel value is pixelValue=10000, then the depth value = pixelValue\*valueScale = 10000\*0.1=1000mm.

Public Member Functions inherited from **ob::VideoFrame**

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **DepthFrame** class, which inherits from the **VideoFrame** class.

Definition at line **501** of file **Frame.hpp**.

## Constructor & Destructor Documentation

- ◆ **DepthFrame()**

```
ob::DepthFrame::DepthFrame ( const ob_frame * impl )  
    inline explicit
```

Construct a new **DepthFrame** object with a given pointer to the internal frame object.

### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

Please use the **FrameFactory** to create a **Frame** object.

### Parameters

**impl** The pointer to the internal frame object.

Definition at line **514** of file **Frame.hpp**.

### ◆ ~DepthFrame()

```
ob::DepthFrame::~DepthFrame ( )  
    override default noexcept
```

## Member Function Documentation

### ◆ getValueScale()

```
float ob::DepthFrame::getValueScale ( ) const  
    inline
```

Get the value scale of the depth frame. The pixel value of depth frame is multiplied by the scale to give a depth value in millimeters. For example, if `valueScale=0.1` and a certain coordinate pixel value is `pixelValue=10000`, then the depth value = `pixelValue*valueScale = 10000*0.1=1000mm`.

### Returns

`float` The scale.

Definition at line **525** of file **Frame.hpp**.

Referenced by **getValueScale()**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Frame.hpp**



## ob::Device Member List

This is the complete list of members for **ob::Device**, including all inherited members.

**Device**(ob\_device\_t \*impl)  
**Device**(Device &&other) noexcept  
**Device**(const Device &)=delete  
**DeviceFwUpdateCallback** typedef  
**deviceStateChangeCallback\_**  
**DeviceStateChangedCallback** typedef  
**deviceStateChangedCallback**(OBDeviceState state, const char \*message, void \*userData)  
**deviceUpgrade**(const char \*filePath, DeviceFwUpdateCallback callback, bool async=true)  
**deviceUpgradeFromData**(const uint8\_t \*firmwareData, uint32\_t firmwareDataSize, DeviceFwUpdateCallback callback)  
**enableGlobalTimestamp**(bool enable)  
**enableHeartbeat**(bool enable) const  
**exportSettingsAsPresetJsonData**(const char \*presetName, const uint8\_t \*\*data, uint32\_t \*dataSize)  
**exportSettingsAsPresetJsonFile**(const char \*filePath) const  
**fwUpdateCallback\_**  
**getAvailableFrameInterleaveList**() const  
**getAvailablePresetList**() const  
**getAvailablePresetResolutionConfigList**() const  
**getBoolProperty**(OBPropertyID propertyId) const  
**getBoolPropertyRange**(OBPropertyID propertyId) const  
**getCalibrationCameraParamList**()  
**getCurrentDepthModeName**()  
**getCurrentDepthWorkMode**() const  
**getCurrentPresetName**() const  
**getDepthWorkModeList**() const  
**getDeviceInfo**() const  
**getDeviceState**()  
**getExtensionInfo**(const std::string &infoKey) const  
**getFloatProperty**(OBPropertyID propertyId) const  
**getFloatPropertyRange**(OBPropertyID propertyId) const  
**getImpl**() const  
**getIntProperty**(OBPropertyID propertyId) const  
**getIntPropertyRange**(OBPropertyID propertyId) const

**getMultiDeviceSyncConfig()** const  
**getSensor**(OBSensorType type) const  
**getSensorList()** const  
**getStructuredData**(OBPropertyID propertyId, uint8\_t \*data, uint32\_t \*dataSize) const  
**getSupportedMultiDeviceSyncModeBitmap()** const  
**getSupportedProperty**(uint32\_t index) const  
**getSupportedPropertyCount()** const  
**getTimestampResetConfig()** const  
**impl\_**  
**isExtensionInfoExist**(const std::string &infoKey) const  
**isFrameInterleaveSupported()** const  
**isGlobalTimestampSupported()** const  
**isPropertySupported**(OBPropertyID propertyId, OBPermissionType permission) const  
**loadDepthFilterConfig**(const char \*filePath)  
**loadFrameInterleave**(const char \*frameInterleaveName) const  
**loadPreset**(const char \*presetName) const  
**loadPresetFromJsonData**(const char \*presetName, const uint8\_t \*data, uint32\_t size)  
**loadPresetFromFile**(const char \*filePath) const  
**operator=**(Device &&other) noexcept  
**operator=**(const Device &) = delete  
**readCustomerData**(void \*data, uint32\_t \*dataSize)  
**reboot()** const  
**reboot**(uint32\_t delayMs) const  
**sendAndReceiveData**(const uint8\_t \*sendData, uint32\_t sendDataSize, uint8\_t \*receiveData, uint32\_t \*receiveDataSize) const  
**setBoolProperty**(OBPropertyID propertyId, bool value) const  
**setDeviceStateChangedCallback**(DeviceStateChangedCallback callback)  
**setFloatProperty**(OBPropertyID propertyId, float value) const  
**setIntProperty**(OBPropertyID propertyId, int32\_t value) const  
**setMultiDeviceSyncConfig**(const OBMultiDeviceSyncConfig &config) const  
**setStructuredData**(OBPropertyID propertyId, const uint8\_t \*data, uint32\_t dataSize) const  
**setTimestampResetConfig**(const OBDeviceTimestampResetConfig &config) const  
**switchDepthWorkMode**(const OBDepthWorkMode &workMode) const  
**switchDepthWorkMode**(const char \*modeName) const  
**timerSyncWithHost()** const  
**timestampReset()** const  
**triggerCapture()** const

```
updateFirmware(const char *filePath, DeviceFwUpdateCallback callback, bool async=true)
updateFirmwareFromData(const uint8_t *firmwareData, uint32_t firmwareDataSize, DeviceFwUpdateCallbac
updateOptionalDepthPresets(const char filePathList[][OB_PATH_MAX], uint8_t pathCount, DeviceFwUpdat
writeCustomerData(const void *data, uint32_t dataSize)

~Device() noexcept
```

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::Device Class Reference

```
#include <Device.hpp>
```

Inheritance diagram for ob::Device:

## Public Types

```
typedef std::function< void(OBFwUpdateState state, const char *message, uint8_t percent)> DeviceFwUpdateCallback;
firmware update.

typedef std::function< void(OBDeviceState state, const char *message)> DeviceStateChangeCallback;
status updates.
```

## Public Member Functions

**Device** (ob\_device\_t \*impl)

Describe the entity of the RGBD camera, representing a specific model of RGBD camera.

**Device** (**Device** &&other) noexcept

**Device** & **operator=** (**Device** &&other) noexcept

**Device** (const **Device** &)=delete

**Device** & **operator=** (const **Device** &)=delete

virtual ~**Device** () noexcept

ob\_device\_t \* **getImpl** () const

std::shared\_ptr<**DeviceInfo**> **getDeviceInfo** () const

Get device information.

bool **isExtensionInfoExist** (const std::string &infoKey)

const

Check if the extension information is exist.

const char \* **getExtensionInfo** (const std::string &infoKey) const

Get information about extensions obtained from SDK supported by the device.

std::shared\_ptr<**SensorList**> **getSensorList** () const

Get device sensor list.

std::shared\_ptr<**Sensor**> **getSensor** (OBSensorType type) const

Get specific type of sensor if device not open, SDK will automatically open the connected device and return to the instance.

```

void setIntProperty (OBPropertyID propertyId, int32_t
    value) const
    Set int type of device property.

void setFloatProperty (OBPropertyID propertyId, float
    value) const
    Set float type of device property.

void setBoolProperty (OBPropertyID propertyId, bool
    value) const
    Set bool type of device property.

int32_t getIntProperty (OBPropertyID propertyId) const
    Get int type of device property.

float getFloatProperty (OBPropertyID propertyId) const
    Get float type of device property.

bool getBoolProperty (OBPropertyID propertyId) const
    Get bool type of device property.

OBIntPropertyRange getIntPropertyRange (OBPropertyID propertyId)
    const
    Get int type device property range (including current
    value and default value)

OBFloatPropertyRange getFloatPropertyRange (OBPropertyID propertyId)
    const
    Get float type device property range((including current
    value and default value)

OBBoolPropertyRange getBoolPropertyRange (OBPropertyID propertyId)
    const
    Get bool type device property range (including current
    value and default value)

void setStructuredData (OBPropertyID propertyId, const
    uint8_t *data, uint32_t dataSize) const
    Set the structured data type of a device property.

void getStructuredData (OBPropertyID propertyId,
    uint8_t *data, uint32_t *dataSize) const
    Get the structured data type of a device property.

void writeCustomerData (const void *data, uint32_t
    dataSize)
    Set the customer data type of a device property.

void readCustomerData (void *data, uint32_t *dataSize)
    Get the customer data type of a device property.

int getSupportedPropertyCount () const

```

Get the number of properties supported by the device.

**OBPropertyItem** **getSupportedProperty** (uint32\_t index) const

Get the supported properties of the device.

bool **isPropertySupported** (**OBPropertyID** propertyId,  
**OBPermissionType** permission) const

Check if a property permission is supported.

bool **isGlobalTimestampSupported** () const

Check if the global timestamp is supported for the  
device.

void **enableGlobalTimestamp** (bool enable)

Enable or disable the global timestamp.

void **updateFirmware** (const char \*filePath,  
**DeviceFwUpdateCallback** callback, bool async=true)  
Update the device firmware.

void **updateFirmwareFromData** (const uint8\_t  
\*firmwareData, uint32\_t firmwareDataSize,  
**DeviceFwUpdateCallback** callback, bool async=true)  
Update the device firmware from data.

void **updateOptionalDepthPresets** (const char  
filePathList[][**OB\_PATH\_MAX**], uint8\_t pathCount,  
**DeviceFwUpdateCallback** callback)  
Update the device optional depth presets.

void **setDeviceStateChangedCallback**  
(**DeviceStateChangedCallback** callback)  
Set the device state changed callbacks.

**OBDepthWorkMode** **getCurrentDepthWorkMode** () const

Get current depth work mode.

const char \* **getCurrentDepthModeName** ()

Get current depth mode name.

**OBStatus** **switchDepthWorkMode** (const **OBDepthWorkMode**  
&workMode) const

Switch depth work mode by **OBDepthWorkMode**.

Prefer invoke switchDepthWorkMode(const char  
\*modeName) to switch depth mode when known the  
complete name of depth work mode.

**OBStatus** **switchDepthWorkMode** (const char \*modeName)  
const

Switch depth work mode by work mode name.

std::shared\_ptr<**OBDepthWorkModeList**> **getDepthWorkModeList** () const

Request support depth work mode list.

void **reboot** () const  
    **Device** restart.

void **reboot** (uint32\_t delayMs) const  
    **Device** restart delay mode.

void **enableHeartbeat** (bool enable) const  
    Enable or disable the device heartbeat.

uint16\_t **getSupportedMultiDeviceSyncModeBitmap** () const  
    Get the supported multi device sync mode bitmap of the device.

void **setMultiDeviceSyncConfig** (const  
    **OBMultiDeviceSyncConfig** &config) const  
    set the multi device sync configuration of the device.

**OBMultiDeviceSyncConfig** **getMultiDeviceSyncConfig** () const  
    get the multi device sync configuration of the device.

void **triggerCapture** () const  
    send the capture command to the device.

void **setTimestampResetConfig** (const  
    **OBDeviceTimestampResetConfig** &config) const  
    set the timestamp reset configuration of the device.

**OBDeviceTimestampResetConfig** **getTimestampResetConfig** () const  
    get the timestamp reset configuration of the device.

void **timestampReset** () const  
    send the timestamp reset command to the device.

void **timerSyncWithHost** () const  
    synchronize the timer of the device with the host.

const char \* **getCurrentPresetName** () const  
    Get current preset name.

void **loadPreset** (const char \*presetName) const  
    load the preset according to the preset name.

std::shared\_ptr<**DevicePresetList**> **getAvailablePresetList** () const  
    Get available preset list.

void **loadPresetFromJsonFile** (const char \*filePath) const  
    Load custom preset from file.

void **loadPresetFromJsonData** (const char \*presetName,  
    const uint8\_t \*data, uint32\_t size)  
    Load custom preset from data.

```

void exportSettingsAsPresetJsonData (const char
    *presetName, const uint8_t **data, uint32_t *dataSize)
    Export current device settings as a preset json data.

void exportSettingsAsPresetJsonFile (const char
    *filePath) const
    Export current device settings as a preset json file.

OBDeviceState getDeviceState ()
    Get the current device status.

void sendAndReceiveData (const uint8_t *sendData,
    uint32_t sendDataSize, uint8_t *receiveData, uint32_t
    *receiveDataSize) const
    Send data to the device and receive data from the
    device.

bool isFrameInterleaveSupported () const
    Check if the device supports the frame interleave
    feature.

void loadFrameInterleave (const char
    *frameInterleaveName) const
    load the frame interleave according to frame interleave
    name.

std::shared_ptr<DeviceFrameInterleaveList> getAvailableFrameInterleaveList () const
    Get available frame interleave list.

std::shared_ptr<PresetResolutionConfigList> getAvailablePresetResolutionConfigList () const
    Get available frame interleave list.

void deviceUpgrade (const char *filePath,
    DeviceFwUpdateCallback callback, bool async=true)
void deviceUpgradeFromData (const uint8_t
    *firmwareData, uint32_t firmwareDataSize,
    DeviceFwUpdateCallback callback, bool async=true)

std::shared_ptr<CameraParamList> getCalibrationCameraParamList ()
void loadDepthFilterConfig (const char *filePath)

```

## Static Public Member Functions

```
static void deviceStateChangedCallback (OBDeviceState state, const char *message, void *userData)
```

## Protected Attributes

```
ob_device * impl_ = nullptr
```

```
DeviceStateChangedCallback deviceStateChangeCallback_
```

## **DeviceFwUpdateCallback fwUpdateCallback\_**

### Detailed Description

Definition at line **35** of file **Device.hpp**.

### Member Typedef Documentation

#### ◆ DeviceFwUpdateCallback

```
typedef std::function<void(OBFwUpdateState state, const char *message, uint8_t percent)>
ob::Device::DeviceFwUpdateCallback
```

Callback function for device firmware update progress.

#### Parameters

- state** The device firmware update status.
- message** Status information.
- percent** The percentage of the update progress.

Definition at line **44** of file **Device.hpp**.

#### ◆ DeviceStateChangedCallback

```
typedef std::function<void(OBDeviceState state, const char *message)>
ob::Device::DeviceStateChangedCallback
```

Callback function for device status updates.

#### Parameters

- state** The device status.
- message** Status information.

Definition at line **52** of file **Device.hpp**.

### Constructor & Destructor Documentation

◆ Device() [1/3]

ob::Device::Device ( ob\_device\_t \* impl ) inline explicit

Describe the entity of the RGBD camera, representing a specific model of RGBD camera.

Definition at line [63](#) of file [Device.hpp](#).

Referenced by [Device\(\)](#), [Device\(\)](#), [deviceStateChangedCallback\(\)](#), [operator=\(\)](#), [operator=\(\)](#), [ob::PlaybackDevice::PlaybackDevice\(\)](#), and [ob::PlaybackDevice::PlaybackDevice\(\)](#).

◆ Device() [2/3]

ob::Device::Device ( **Device** && other ) inline noexcept

Definition at line [65](#) of file [Device.hpp](#).

◆ Device() [3/3]

ob::Device::Device ( const **Device** & ) delete

◆ ~Device()

virtual ob::Device::~Device ( ) inline virtual noexcept

Definition at line [83](#) of file [Device.hpp](#).

## Member Function Documentation

◆ operator=() [1/2]

**Device** & ob::Device::operator= ( **Device** && other ) inline noexcept

Definition at line [69](#) of file [Device.hpp](#).

Referenced by [ob::PlaybackDevice::operator=\(\)](#).

◆ operator=()

**Device** & ob::Device::operator= ( const **Device** & )

delete

◆ getImpl()

ob\_device\_t \* ob::Device::getImpl ( ) const

inline

Definition at line **89** of file **Device.hpp**.

◆ getDeviceInfo()

std::shared\_ptr<**DeviceInfo**> ob::Device::getDeviceInfo ( ) const

inline

Get device information.

**Returns**

std::shared\_ptr<DeviceInfo> return device information

Definition at line **98** of file **Device.hpp**.

◆ isExtensionInfoExist()

bool ob::Device::isExtensionInfoExist ( const std::string & infoKey ) const

inline

Check if the extension information is exist.

**Parameters**

**infoKey** The key of the extension information

**Returns**

bool Whether the extension information exists

Definition at line **111** of file **Device.hpp**.

◆ getExtensionInfo()

```
const char* ob::Device::getExtensionInfo ( const std::string & infoKey ) const
```

inline

Get information about extensions obtained from SDK supported by the device.

#### Parameters

**infoKey** The key of the extension information

#### Returns

const char\* Returns extended information about the device

Definition at line [124](#) of file [Device.hpp](#).

### ◆ [getSensorList\(\)](#)

```
std::shared_ptr<SensorList> ob::Device::getSensorList ( ) const
```

inline

Get device sensor list.

#### Returns

std::shared\_ptr<SensorList> return the sensor list

Definition at line [136](#) of file [Device.hpp](#).

### ◆ [getSensor\(\)](#)

```
std::shared_ptr<Sensor> ob::Device::getSensor ( OBSTensorType type ) const
```

inline

Get specific type of sensor if device not open, SDK will automatically open the connected device and return to the instance.

#### Returns

std::shared\_ptr<Sensor> return the sensor example, if the device does not have the device,return nullptr

Definition at line [149](#) of file [Device.hpp](#).

### ◆ [setIntProperty\(\)](#)

```
void ob::Device::setIntProperty ( OBPropertyID propertyId,  
                                int32_t           value ) const  
                                         inline
```

Set int type of device property.

#### Parameters

**propertyId** Property id  
**value** Property value to be set

Definition at line [162](#) of file [Device.hpp](#).

Referenced by [reboot\(\)](#).

#### ◆ [setFloatProperty\(\)](#)

```
void ob::Device::setFloatProperty ( OBPropertyID propertyId,  
                                    float            value ) const  
                                         inline
```

Set float type of device property.

#### Parameters

**propertyId** Property id  
**value** Property value to be set

Definition at line [174](#) of file [Device.hpp](#).

#### ◆ [setBoolProperty\(\)](#)

```
void ob::Device::setBoolProperty ( OBPropertyID propertyId,  
                                   bool             value ) const  
                                         inline
```

Set bool type of device property.

#### Parameters

**propertyId** Property id  
**value** Property value to be set

Definition at line [186](#) of file [Device.hpp](#).

#### ◆ [getIntProperty\(\)](#)

```
int32_t ob::Device::getIntProperty ( OBPropertyID propertyId ) const
```

inline

Get int type of device property.

#### Parameters

**propertyId** Property id

#### Returns

int32\_t Property to get

Definition at line **198** of file [Device.hpp](#).

### ◆ [getFloatProperty\(\)](#)

```
float ob::Device::getFloatProperty ( OBPropertyID propertyId ) const
```

inline

Get float type of device property.

#### Parameters

**propertyId** Property id

#### Returns

float Property to get

Definition at line **211** of file [Device.hpp](#).

### ◆ [getBoolProperty\(\)](#)

```
bool ob::Device::getBoolProperty ( OBPropertyID propertyId ) const
```

inline

Get bool type of device property.

#### Parameters

**propertyId** Property id

#### Returns

bool Property to get

Definition at line **224** of file [Device.hpp](#).

### ◆ [getIntPropertyRange\(\)](#)

**OBIntPropertyRange** ob::Device::getIntPropertyRange ( **OBPropertyID** propertyId ) const inline

Get int type device property range (including current value and default value)

#### Parameters

**propertyId** Property id

#### Returns

**OBIntPropertyRange** Property range

Definition at line [237](#) of file **Device.hpp**.

◆ [getFloatPropertyRange\(\)](#)

**OBFloatPropertyRange** ob::Device::getFloatPropertyRange ( **OBPropertyID** propertyId ) const inline

Get float type device property range((including current value and default value)

#### Parameters

**propertyId** Property id

#### Returns

**OBFloatPropertyRange** Property range

Definition at line [250](#) of file **Device.hpp**.

◆ [getBoolPropertyRange\(\)](#)

**OBBoolPropertyRange** ob::Device::getBoolPropertyRange ( **OBPropertyID** propertyId ) const inline

Get bool type device property range (including current value and default value)

#### Parameters

**propertyId** The ID of the property

#### Returns

**OBBoolPropertyRange** The range of the property

Definition at line [263](#) of file **Device.hpp**.

◆ [setStructuredData\(\)](#)

```
void ob::Device::setStructuredData ( OBPropertyID propertyId,  
                                    const uint8_t * data,  
                                    uint32_t      dataSize ) const  
    inline
```

Set the structured data type of a device property.

### Parameters

- propertyId** The ID of the property
- data** The data to set
- dataSize** The size of the data to set

Definition at line [277](#) of file [Device.hpp](#).

### ◆ [getStructuredData\(\)](#)

```
void ob::Device::getStructuredData ( OBPropertyID propertyId,  
                                    uint8_t *      data,  
                                    uint32_t *     dataSize ) const  
    inline
```

Get the structured data type of a device property.

### Parameters

- propertyId** The ID of the property
- data** The property data obtained
- dataSize** The size of the data obtained

Definition at line [290](#) of file [Device.hpp](#).

### ◆ [writeCustomerData\(\)](#)

```
void ob::Device::writeCustomerData ( const void * data,
                                    uint32_t      dataSize )
```

inline

Set the customer data type of a device property.

#### Parameters

**data** The data to set

**dataSize** The size of the data to set, the maximum length cannot exceed 65532 bytes.

Definition at line [302](#) of file [Device.hpp](#).

#### ◆ [readCustomerData\(\)](#)

```
void ob::Device::readCustomerData ( void *      data,
                                    uint32_t *   dataSize )
```

inline

Get the customer data type of a device property.

#### Parameters

**data** The property data obtained

**dataSize** The size of the data obtained

Definition at line [314](#) of file [Device.hpp](#).

#### ◆ [getSupportedPropertyCount\(\)](#)

```
int ob::Device::getSupportedPropertyCount ( ) const
```

inline

Get the number of properties supported by the device.

#### Returns

The number of supported properties

Definition at line [325](#) of file [Device.hpp](#).

#### ◆ [getSupportedProperty\(\)](#)

**OBPropertyItem** ob::Device::getSupportedProperty ( uint32\_t index ) const

inline

Get the supported properties of the device.

#### Parameters

**index** The index of the property

#### Returns

The type of supported property

Definition at line [338](#) of file **Device.hpp**.

### ◆ isPropertySupported()

bool ob::Device::isPropertySupported ( **OBPropertyID** propertyId,  
**OBPermissionType** permission ) const

inline

Check if a property permission is supported.

#### Parameters

**propertyId** The ID of the property

**permission** The read and write permissions to check

#### Returns

Whether the property permission is supported

Definition at line [352](#) of file **Device.hpp**.

### ◆ isGlobalTimestampSupported()

bool ob::Device::isGlobalTimestampSupported ( ) const

inline

Check if the global timestamp is supported for the device.

#### Returns

Whether the global timestamp is supported

Definition at line [364](#) of file **Device.hpp**.

### ◆ enableGlobalTimestamp()

```
void ob::Device::enableGlobalTimestamp ( bool enable )
```

inline

Enable or disable the global timestamp.

### Parameters

**enable** Whether to enable the global timestamp

Definition at line [376](#) of file [Device.hpp](#).

## ◆ updateFirmware()

```
void ob::Device::updateFirmware ( const char * filePath,  
                                DeviceFwUpdateCallback callback,  
                                bool async = true )
```

inline

Update the device firmware.

### Parameters

**filePath** Firmware path

**callback** Firmware Update progress and status callback

**async** Whether to execute asynchronously

Definition at line [389](#) of file [Device.hpp](#).

Referenced by [deviceUpgrade\(\)](#).

## ◆ updateFirmwareFromData()

```
void ob::Device::updateFirmwareFromData ( const uint8_t * firmwareData,
                                         uint32_t firmwareDataSize,
                                         DeviceFwUpdateCallback callback,
                                         bool async = true ) inline
```

Update the device firmware from data.

#### Parameters

**firmwareData** Firmware data  
**firmwareDataSize** Firmware data size  
**callback** Firmware Update progress and status callback  
**async** Whether to execute asynchronously

Definition at line [404](#) of file [Device.hpp](#).

Referenced by [deviceUpgradeFromData\(\)](#).

#### ◆ updateOptionalDepthPresets()

```
void ob::Device::updateOptionalDepthPresets ( const char filePathList[][OB_PATH_MAX],
                                              uint8_t pathCount,
                                              DeviceFwUpdateCallback callback ) inline
```

Update the device optional depth presets.

#### Parameters

**filePathList** A list(2D array) of preset file paths, each up to OB\_PATH\_MAX characters.  
**pathCount** The number of the preset file paths.  
**callback** Preset update progress and status callback

Definition at line [418](#) of file [Device.hpp](#).

#### ◆ setDeviceStateChangedCallback()

```
void ob::Device::setDeviceStateChangedCallback ( DeviceStateChangedCallback callback )           inline
```

Set the device state changed callbacks.

#### Parameters

**callback** The callback function that is triggered when the device status changes (for example, the frame rate is automatically reduced or the stream is closed due to high temperature, etc.)

Definition at line [431](#) of file **Device.hpp**.

#### ◆ **deviceStateChangedCallback()**

```
void ob::Device::deviceStateChangedCallback ( OBDeviceState state,  
                                              const char *      message,  
                                              void *           userData )           inline static
```

Definition at line [438](#) of file **Device.hpp**.

Referenced by **setDeviceStateChangedCallback()**.

#### ◆ **getCurrentDepthWorkMode()**

```
OBDepthWorkMode ob::Device::getCurrentDepthWorkMode ( ) const           inline
```

Get current depth work mode.

#### Returns

**ob\_depth\_work\_mode** Current depth work mode

Definition at line [448](#) of file **Device.hpp**.

#### ◆ **getCurrentDepthModeName()**

```
const char * ob::Device::getCurrentDepthModeName ( )
```

inline

Get current depth mode name.

According the current preset name to return current depth mode name

### Returns

const char\* return the current depth mode name.

Definition at line **460** of file [Device.hpp](#).

### ◆ switchDepthWorkMode() [1/2]

**OBStatus** ob::Device::switchDepthWorkMode ( const **OBDepthWorkMode** & workMode ) const

inline

Switch depth work mode by **OBDepthWorkMode**. Prefer invoke switchDepthWorkMode(const char \*modeName) to switch depth mode when known the complete name of depth work mode.

### Parameters

[in] **workMode** Depth work mode come from **ob\_depth\_work\_mode\_list** which return by  
ob\_device\_get\_depth\_work\_mode\_list

Definition at line **472** of file [Device.hpp](#).

### ◆ switchDepthWorkMode() [2/2]

**OBStatus** ob::Device::switchDepthWorkMode ( const char \* modeName ) const

inline

Switch depth work mode by work mode name.

### Parameters

[in] **modeName** Depth work mode name which equals to **OBDepthWorkMode.name**

Definition at line **484** of file [Device.hpp](#).

### ◆ getDepthWorkModeList()

```
std::shared_ptr<OBDepthWorkModeList> ob::Device::getDepthWorkModeList( ) const inline
```

Request support depth work mode list.

#### Returns

**OBDepthWorkModeList** list of **ob\_depth\_work\_mode**

Definition at line **495** of file **Device.hpp**.

#### ◆ reboot()

```
void ob::Device::reboot( ) const
```

inline

**Device** restart.

#### Attention

The device will be disconnected and reconnected. After the device is disconnected, the access to the **Device** object interface may be abnormal. Please delete the object directly and obtain it again after the device is reconnected.

Definition at line **507** of file **Device.hpp**.

Referenced by **reboot()**.

#### ◆ reboot()

```
void ob::Device::reboot( uint32_t delayMs ) const
```

inline

**Device** restart delay mode.

#### Attention

The device will be disconnected and reconnected. After the device is disconnected, the access to the **Device** object interface may be abnormal. Please delete the object directly and obtain it again after the device is reconnected. Support devices: Gemini2 L

#### Parameters

[in] **delayMs** Time unit: ms. delayMs == 0: No delay; delayMs > 0, Delay millisecond connect to host device after reboot

Definition at line **521** of file **Device.hpp**.

## ◆ enableHeartbeat()

void ob::Device::enableHeartbeat ( bool enable ) const

inline

Enable or disable the device heartbeat.

After enable the device heartbeat, the sdk will start a thread to send heartbeat signal to the device error every 3 seconds.

### Attention

If the device does not receive the heartbeat signal for a long time, it will be disconnected and rebooted.

### Parameters

[in] **enable** Whether to enable the device heartbeat.

Definition at line [534](#) of file [Device.hpp](#).

## ◆ getSupportedMultiDeviceSyncModeBitmap()

uint16\_t ob::Device::getSupportedMultiDeviceSyncModeBitmap ( ) const

inline

Get the supported multi device sync mode bitmap of the device.

For example, if the return value is 0b00001100, it means the device supports

**OB\_MULTI\_DEVICE\_SYNC\_MODE\_PRIMARY** and

**OB\_MULTI\_DEVICE\_SYNC\_MODE\_SECONDARY**. User can check the supported mode by the code:

```
if(supported_mode_bitmap & OB_MULTI_DEVICE_SYNC_MODE_FREE_RUN){  
    //support OB_MULTI_DEVICE_SYNC_MODE_FREE_RUN  
}  
if(supported_mode_bitmap & OB_MULTI_DEVICE_SYNC_MODE_STANDALONE){  
    //support OB_MULTI_DEVICE_SYNC_MODE_STANDALONE  
}  
// and so on
```

### Returns

uint16\_t return the supported multi device sync mode bitmap of the device.

Definition at line [555](#) of file [Device.hpp](#).

## ◆ setMultiDeviceSyncConfig()

```
void ob::Device::setMultiDeviceSyncConfig ( const OBMultiDeviceSyncConfig & config ) const inline
```

set the multi device sync configuration of the device.

### Parameters

[in] **config** The multi device sync configuration.

Definition at line [567](#) of file **Device.hpp**.

### ◆ **getMultiDeviceSyncConfig()**

```
OBMultiDeviceSyncConfig ob::Device::getMultiDeviceSyncConfig ( ) const inline
```

get the multi device sync configuration of the device.

### Returns

**OBMultiDeviceSyncConfig** return the multi device sync configuration of the device.

Definition at line [578](#) of file **Device.hpp**.

### ◆ **triggerCapture()**

```
void ob::Device::triggerCapture ( ) const inline
```

send the capture command to the device.

The device will start one time image capture after receiving the capture command when it is in the

### **OB\_MULTI\_DEVICE\_SYNC\_MODE\_SOFTWARE\_TRIGGERING**

#### Attention

The frequency of the user call this function multiplied by the number of frames per trigger should be less than the frame rate of the stream. The number of frames per trigger can be set by `framesPerTrigger`.

For some models, receive and execute the capture command will have a certain delay and performance consumption, so the frequency of calling this function should not be too high, please refer to the product manual for the specific supported frequency.

If the device is not in the

**OB\_MULTI\_DEVICE\_SYNC\_MODE\_HARDWARE\_TRIGGERING** mode, device will ignore the capture command.

Definition at line [596](#) of file **Device.hpp**.

◆ **setTimestampResetConfig()**

void ob::Device::setTimestampResetConfig ( const **OBDeviceTimestampResetConfig** & config ) const      inline

set the timestamp reset configuration of the device.

Definition at line **605** of file **Device.hpp**.

◆ **getTimestampResetConfig()**

**OBDeviceTimestampResetConfig** ob::Device::getTimestampResetConfig ( ) const      inline

get the timestamp reset configuration of the device.

**Returns**

**OBDeviceTimestampResetConfig** return the timestamp reset configuration of the device.

Definition at line **616** of file **Device.hpp**.

◆ **timestampReset()**

void ob::Device::timestampReset ( ) const      inline

send the timestamp reset command to the device.

The device will reset the timer for calculating the timestamp for output frames to 0 after receiving the timestamp reset command when the timestamp reset function is enabled. The timestamp reset function can be enabled by call **ob\_device\_set\_timestamp\_reset\_config**.

Before calling this function, user should call **ob\_device\_set\_timestamp\_reset\_config** to disable the timestamp reset function (It is not required for some models, but it is still recommended to do so for code compatibility).

**Attention**

If the stream of the device is started, the timestamp of the continuous frames output by the stream will jump once after the timestamp reset.

Due to the timer of device is not high-accuracy, the timestamp of the continuous frames output by the stream will drift after a long time. User can call this function periodically to reset the timer to avoid the timestamp drift, the recommended interval time is 60 minutes.

Definition at line **634** of file **Device.hpp**.

## ◆ timerSyncWithHost()

void ob::Device::timerSyncWithHost ( ) const

inline

synchronize the timer of the device with the host.

After calling this function, the timer of the device will be synchronized with the host. User can call this function to multiple devices to synchronize all timers of the devices.

### Attention

If the stream of the device is started, the timestamp of the continuous frames output by the stream will may jump once after the timer sync.

Due to the timer of device is not high-accuracy, the timestamp of the continuous frames output by the stream will drift after a long time. User can call this function periodically to synchronize the timer to avoid the timestamp drift, the recommended interval time is 60 minutes.

Definition at line [656](#) of file [Device.hpp](#).

## ◆ getCurrentPresetName()

const char \* ob::Device::getCurrentPresetName ( ) const

inline

Get current preset name.

The preset mean a set of parameters or configurations that can be applied to the device to achieve a specific effect or function.

### Returns

const char\* return the current preset name, it should be one of the preset names returned by [getAvailablePresetList](#).

Definition at line [667](#) of file [Device.hpp](#).

## ◆ loadPreset()

```
void ob::Device::loadPreset ( const char * presetName ) const
```

inline

load the preset according to the preset name.

### Attention

After loading the preset, the settings in the preset will set to the device immediately. Therefore, it is recommended to re-read the device settings to update the user program temporarily.

### Parameters

**presetName** The preset name to set. The name should be one of the preset names returned by [getAvailablePresetList](#).

Definition at line [680](#) of file [Device.hpp](#).

### ◆ [getAvailablePresetList\(\)](#)

```
std::shared_ptr<DevicePresetList> ob::Device::getAvailablePresetList ( ) const
```

inline

Get available preset list.

The available preset list usually defined by the device manufacturer and restores on the device.

User can load the custom preset by calling [loadPresetFromFile](#) to append the available preset list.

### Returns

[DevicePresetList](#) return the available preset list.

Definition at line [693](#) of file [Device.hpp](#).

### ◆ [loadPresetFromFile\(\)](#)

```
void ob::Device::loadPresetFromFile ( const char * filePath ) const inline
```

Load custom preset from file.

After loading the custom preset, the settings in the custom preset will set to the device immediately.

After loading the custom preset, the available preset list will be appended with the custom preset and named as the file name.

### Attention

The user should ensure that the custom preset file is adapted to the device and the settings in the file are valid.

It is recommended to re-read the device settings to update the user program temporarily after successfully loading the custom preset.

### Parameters

**filePath** The path of the custom preset file.

Definition at line [710](#) of file [Device.hpp](#).

◆ [loadPresetFromJsonData\(\)](#)

```
void ob::Device::loadPresetFromJsonData ( const char * presetName,  
                                         const uint8_t * data,  
                                         uint32_t size )  
                                         inline
```

Load custom preset from data.

After loading the custom preset, the settings in the custom preset will set to the device immediately.

After loading the custom preset, the available preset list will be appended with the custom preset and named as the presetName.

### Attention

The user should ensure that the custom preset data is adapted to the device and the settings in the data are valid.

It is recommended to re-read the device settings to update the user program temporarily after successfully loading the custom preset.

### Parameters

**data** The custom preset data.

**size** The size of the custom preset data.

Definition at line [727](#) of file [Device.hpp](#).

◆ [exportSettingsAsPresetJsonData\(\)](#)

```
void ob::Device::exportSettingsAsPresetJsonData ( const char *      presetName,
                                                const uint8_t **  data,
                                                uint32_t *       dataSize )           inline
```

Export current device settings as a preset json data.

After exporting the preset, a new preset named as the presetName will be added to the available preset list.

### Attention

The memory of the data is allocated by the SDK, and will automatically be released by the SDK.  
The memory of the data will be reused by the SDK on the next call, so the user should copy the data to a new buffer if it needs to be preserved.

### Parameters

[out] **data** return the preset json data.

[out] **dataSize** return the size of the preset json data.

Definition at line [743](#) of file [Device.hpp](#).

### ◆ [exportSettingsAsPresetJsonFile\(\)](#)

```
void ob::Device::exportSettingsAsPresetJsonFile ( const char * filePath ) const           inline
```

Export current device settings as a preset json file.

The exported preset file can be loaded by calling [loadPresetFromJsonFile](#) to restore the device setting.

After exporting the preset, a new preset named as the filePath will be added to the available preset list.

### Parameters

**filePath** The path of the preset file to be exported.

Definition at line [755](#) of file [Device.hpp](#).

### ◆ [getDeviceState\(\)](#)

**OBDeviceState** ob::Device::getDeviceState ( )

inline

Get the current device status.

### Returns

**OBDeviceState** The device state information.

Definition at line **766** of file **Device.hpp**.

### ◆ sendAndReceiveData()

```
void ob::Device::sendAndReceiveData ( const uint8_t * sendData,
                                      uint32_t      sendDataSet,
                                      uint8_t *      receiveData,
                                      uint32_t *     receiveDataSet ) const
```

inline

Send data to the device and receive data from the device.

This is a factory and debug function, which can be used to send and receive data from the device. The data format is secret and belongs to the device vendor.

### Attention

The send and receive data buffer are managed by the caller, the receive data buffer should be allocated at 1024 bytes or larger.

### Parameters

- [in] **sendData** The data to be sent to the device.
- [in] **sendDataSet** The size of the data to be sent to the device.
- [out] **receiveData** The data received from the device.
- [in,out] **receiveDataSet** The requested size of the data received from the device, and the actual size of the data received from the device.

Definition at line **785** of file **Device.hpp**.

### ◆ isFrameInterleaveSupported()

```
bool ob::Device::isFrameInterleaveSupported ( ) const inline
```

Check if the device supports the frame interleave feature.

### Returns

bool Returns true if the device supports the frame interleave feature.

Definition at line [796](#) of file [Device.hpp](#).

### ◆ loadFrameInterleave()

```
void ob::Device::loadFrameInterleave ( const char * frameInterleaveName ) const inline
```

load the frame interleave according to frame interleave name.

### Parameters

**frameInterleaveName** The frame interleave name to set. The name should be one of the frame interleave names returned by [getAvailableFrameInterleaveList](#).

Definition at line [808](#) of file [Device.hpp](#).

### ◆ getAvailableFrameInterleaveList()

```
std::shared_ptr<DeviceFrameInterleaveList> ob::Device::getAvailableFrameInterleaveList ( ) const inline
```

Get available frame interleave list.

### Returns

[DeviceFrameInterleaveList](#) return the available frame interleave list.

Definition at line [819](#) of file [Device.hpp](#).

### ◆ getAvailablePresetResolutionConfigList()

```
std::shared_ptr<PresetResolutionConfigList>
ob::Device::getAvailablePresetResolutionConfigList ( ) const inline
```

Get available frame interleave list.

### Returns

**DeviceFrameInterleaveList** return the available frame interleave list.

Definition at line [831](#) of file **Device.hpp**.

### ◆ deviceUpgrade()

```
void ob::Device::deviceUpgrade ( const char * filePath,
                                DeviceFwUpdateCallback callback,
                                bool async = true ) inline
```

Definition at line [848](#) of file **Device.hpp**.

### ◆ deviceUpgradeFromData()

```
void ob::Device::deviceUpgradeFromData ( const uint8_t * firmwareData,
                                         uint32_t firmwareDataSize,
                                         DeviceFwUpdateCallback callback,
                                         bool async = true ) inline
```

Definition at line [852](#) of file **Device.hpp**.

### ◆ getCalibrationCameraParamList()

```
std::shared_ptr<CameraParamList> ob::Device::getCalibrationCameraParamList ( ) inline
```

Definition at line [856](#) of file **Device.hpp**.

### ◆ loadDepthFilterConfig()

```
void ob::Device::loadDepthFilterConfig ( const char * filePath )
```

inline

Definition at line [863](#) of file [Device.hpp](#).

## Member Data Documentation

### ◆ `impl_`

```
ob_device* ob::Device::impl_ = nullptr
```

protected

Definition at line [55](#) of file [Device.hpp](#).

Referenced by [Device\(\)](#), [enableGlobalTimestamp\(\)](#), [enableHeartbeat\(\)](#), [exportSettingsAsPresetJsonData\(\)](#), [exportSettingsAsPresetJsonFile\(\)](#), [getAvailableFrameInterleaveList\(\)](#), [getAvailablePresetList\(\)](#), [getAvailablePresetResolutionConfigList\(\)](#), [getBoolProperty\(\)](#), [getBoolPropertyRange\(\)](#), [getCalibrationCameraParamList\(\)](#), [getCurrentDepthModeName\(\)](#), [getCurrentDepthWorkMode\(\)](#), [getCurrentPresetName\(\)](#), [getDepthWorkModeList\(\)](#), [getDeviceInfo\(\)](#), [getDeviceState\(\)](#), [ob::PlaybackDevice::getDuration\(\)](#), [getExtensionInfo\(\)](#), [getFloatProperty\(\)](#), [getFloatPropertyRange\(\)](#), [getImpl\(\)](#), [getIntProperty\(\)](#), [getIntPropertyRange\(\)](#), [getMultiDeviceSyncConfig\(\)](#), [ob::PlaybackDevice::getPlaybackStatus\(\)](#), [ob::PlaybackDevice::getPosition\(\)](#), [getSensor\(\)](#), [getSensorList\(\)](#), [getStructuredData\(\)](#), [getSupportedMultiDeviceSyncModeBitmap\(\)](#), [getSupportedProperty\(\)](#), [getSupportedPropertyCount\(\)](#), [getTimestampResetConfig\(\)](#), [isExtensionInfoExist\(\)](#), [isFrameInterleaveSupported\(\)](#), [isGlobalTimestampSupported\(\)](#), [isPropertySupported\(\)](#), [loadFrameInterleave\(\)](#), [loadPreset\(\)](#), [loadPresetFromJsonData\(\)](#), [loadPresetFromJsonFile\(\)](#), [operator=\(\)](#), [ob::PlaybackDevice::pause\(\)](#), [ob::PlaybackDevice::PlaybackDevice\(\)](#), [readCustomerData\(\)](#), [reboot\(\)](#), [ob::PlaybackDevice::resume\(\)](#), [ob::PlaybackDevice::seek\(\)](#), [sendAndReceiveData\(\)](#), [setBoolProperty\(\)](#), [setDeviceStateChangedCallback\(\)](#), [setFloatProperty\(\)](#), [setIntProperty\(\)](#), [setMultiDeviceSyncConfig\(\)](#), [ob::PlaybackDevice::setPlaybackRate\(\)](#), [ob::PlaybackDevice::setPlaybackStatusChangeCallback\(\)](#), [setStructuredData\(\)](#), [setTimestampResetConfig\(\)](#), [switchDepthWorkMode\(\)](#), [switchDepthWorkMode\(\)](#), [timerSyncWithHost\(\)](#), [timestampReset\(\)](#), [triggerCapture\(\)](#), [updateFirmware\(\)](#), [updateFirmwareFromData\(\)](#), [updateOptionalDepthPresets\(\)](#), [writeCustomerData\(\)](#), and [~Device\(\)](#).

### ◆ `deviceStateChangeCallback_`

**DeviceStateChangedCallback** ob::Device::deviceStateChangeCallback\_

protected

Definition at line **56** of file **Device.hpp**.

Referenced by **setDeviceStateChangedCallback()**.

◆ fwUpdateCallback\_

**DeviceFwUpdateCallback** ob::Device::fwUpdateCallback\_

protected

Definition at line **57** of file **Device.hpp**.

Referenced by **updateFirmware()**, **updateFirmwareFromData()**, and **updateOptionalDepthPresets()**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Device.hpp**

## ob::DeviceFrameInterleaveList Member List

This is the complete list of members for **ob::DeviceFrameInterleaveList**, including all inherited members.

<b>DeviceFrameInterleaveList</b> (ob_device_frame_interleave_list_t *impl)	<b>ob::DeviceFrameInterleaveList</b>	inlin
<b>getCount()</b>	<b>ob::DeviceFrameInterleaveList</b>	inlin
<b>getName</b> (uint32_t index)	<b>ob::DeviceFrameInterleaveList</b>	inlin
<b>hasFrameInterleave</b> (const char *name)	<b>ob::DeviceFrameInterleaveList</b>	inlin
<b>~DeviceFrameInterleaveList()</b> noexcept	<b>ob::DeviceFrameInterleaveList</b>	inlin

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::DeviceFrameInterleaveList Class Reference

Class representing a list of device **Frame** Interleave. [More...](#)

```
#include <Device.hpp>
```

## Public Member Functions

**DeviceFrameInterleaveList** (ob\_device\_frame\_interleave\_list\_t \*impl)

**~DeviceFrameInterleaveList** () noexcept

**uint32\_t getCount** ()

Get the number of device frame interleave in the list.

**const char \* getName** (uint32\_t index)

Get the name of the device frame interleave at the specified index.

**bool hasFrameInterleave** (const char \*name)

check if the frame interleave list contains the special name frame interleave.

## Detailed Description

Class representing a list of device **Frame** Interleave.

Definition at line [1468](#) of file [Device.hpp](#).

## Constructor & Destructor Documentation

### ◆ DeviceFrameInterleaveList()

```
ob::DeviceFrameInterleaveList::DeviceFrameInterleaveList ( ob_device_frame_interleave_list_t * impl )   inline explicit
```

Definition at line [1473](#) of file [Device.hpp](#).

### ◆ ~DeviceFrameInterleaveList()

```
ob::DeviceFrameInterleaveList::~DeviceFrameInterleaveList ( )   inline noexcept
```

Definition at line [1474](#) of file [Device.hpp](#).

# Member Function Documentation

## ◆ getCount()

`uint32_t ob::DeviceFrameInterleaveList::getCount ( )`

inline

Get the number of device frame interleave in the list.

### Returns

`uint32_t` the number of device frame interleave in the list

Definition at line [1485](#) of file [Device.hpp](#).

## ◆ getName()

`const char * ob::DeviceFrameInterleaveList::getName ( uint32_t index )`

inline

Get the name of the device frame interleave at the specified index.

### Parameters

`index` the index of the device frame interleave

### Returns

`const char*` the name of the device frame interleave

Definition at line [1498](#) of file [Device.hpp](#).

## ◆ hasFrameInterleave()

`bool ob::DeviceFrameInterleaveList::hasFrameInterleave ( const char * name )`

inline

check if the frame interleave list contains the special name frame interleave.

### Parameters

`name` The name of the frame interleave

### Returns

`bool` Returns true if the special name is found in the frame interleave list, otherwise returns false.

Definition at line [1510](#) of file [Device.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Device.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::DeviceInfo Member List

This is the complete list of members for **ob::DeviceInfo**, including all inherited members.

<b>asicName()</b> const	<b>ob::DeviceInfo</b>	inline
<b>connectionType()</b> const	<b>ob::DeviceInfo</b>	inline
<b>DeviceInfo(ob_device_info_t *impl)</b>	<b>ob::DeviceInfo</b>	inline explicit
<b>deviceType()</b> const	<b>ob::DeviceInfo</b>	inline
<b>firmwareVersion()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getAsicName()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getConnectionType()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getDeviceType()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getFirmwareVersion()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getHardwareVersion()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getIpAddress()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getName()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getPid()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getSerialNumber()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getSupportedMinSdkVersion()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getUid()</b> const	<b>ob::DeviceInfo</b>	inline
<b>getVid()</b> const	<b>ob::DeviceInfo</b>	inline
<b>hardwareVersion()</b> const	<b>ob::DeviceInfo</b>	inline
<b>ipAddress()</b> const	<b>ob::DeviceInfo</b>	inline
<b>name()</b> const	<b>ob::DeviceInfo</b>	inline
<b>pid()</b> const	<b>ob::DeviceInfo</b>	inline
<b>serialNumber()</b> const	<b>ob::DeviceInfo</b>	inline
<b>supportedMinSdkVersion()</b> const	<b>ob::DeviceInfo</b>	inline
<b>uid()</b> const	<b>ob::DeviceInfo</b>	inline
<b>vid()</b> const	<b>ob::DeviceInfo</b>	inline
<b>~DeviceInfo()</b> noexcept	<b>ob::DeviceInfo</b>	inline

# ob::DeviceInfo Class Reference

A class describing device information, representing the name, id, serial number and other basic information of an RGBD camera. [More...](#)

```
#include <Device.hpp>
```

## Public Member Functions

```
    DeviceInfo (ob_device_info_t *impl)  
    ~DeviceInfo () noexcept  
const char * getName () const  
    Get device name.  
int getPid () const  
    Get the pid of the device.  
int getVid () const  
    Get the vid of the device.  
const char * getUid () const  
    Get system assigned uid for distinguishing between different devices.  
const char * getSerialNumber () const  
    Get the serial number of the device.  
const char * getFirmwareVersion () const  
    Get the version number of the firmware.  
const char * getConnectionType () const  
    Get the connection type of the device.  
const char * getIpAddress () const  
    Get the IP address of the device.  
const char * getHardwareVersion () const  
    Get the version number of the hardware.  
const char * getSupportedMinSdkVersion () const  
    Get the minimum version number of the SDK supported by the device.  
const char * getAsicName () const  
    Get chip type name.  
OBDeviceType getDeviceType () const  
    Get the device type.  
const char * name () const  
int pid () const
```

```
int vid () const  
const char * uid () const  
const char * serialNumber () const  
const char * firmwareVersion () const  
const char * connectionType () const  
const char * ipAddress () const  
const char * hardwareVersion () const  
const char * supportedMinSdkVersion () const  
const char * asicName () const  
OBDeviceType deviceType () const
```

## Detailed Description

A class describing device information, representing the name, id, serial number and other basic information of an RGBD camera.

Definition at line [872](#) of file [Device.hpp](#).

## Constructor & Destructor Documentation

### ◆ DeviceInfo()

```
ob::DeviceInfo::DeviceInfo ( ob_device_info_t * impl )inline explicit
```

Definition at line [877](#) of file [Device.hpp](#).

### ◆ ~DeviceInfo()

```
ob::DeviceInfo::~DeviceInfo ( )inline noexcept
```

Definition at line [878](#) of file [Device.hpp](#).

## Member Function Documentation

### ◆ getName()

```
const char * ob::DeviceInfo::getName ( ) const
```

inline

Get device name.

**Returns**

```
const char * return the device name
```

Definition at line [889](#) of file [Device.hpp](#).

Referenced by [name\(\)](#).

◆ [getPid\(\)](#)

```
int ob::DeviceInfo::getPid ( ) const
```

inline

Get the pid of the device.

**Returns**

```
int return the pid of the device
```

Definition at line [901](#) of file [Device.hpp](#).

Referenced by [pid\(\)](#).

◆ [getVid\(\)](#)

```
int ob::DeviceInfo::getVid ( ) const
```

inline

Get the vid of the device.

**Returns**

```
int return the vid of the device
```

Definition at line [913](#) of file [Device.hpp](#).

Referenced by [vid\(\)](#).

◆ [getUid\(\)](#)

```
const char * ob::DeviceInfo::getUid ( ) const
```

inline

Get system assigned uid for distinguishing between different devices.

#### Returns

```
const char * return the uid of the device
```

Definition at line [925](#) of file [Device.hpp](#).

Referenced by [uid\(\)](#).

#### ◆ [getSerialNumber\(\)](#)

```
const char * ob::DeviceInfo::getSerialNumber ( ) const
```

inline

Get the serial number of the device.

#### Returns

```
const char * return the serial number of the device
```

Definition at line [937](#) of file [Device.hpp](#).

Referenced by [serialNumber\(\)](#).

#### ◆ [getFirmwareVersion\(\)](#)

```
const char * ob::DeviceInfo::getFirmwareVersion ( ) const
```

inline

Get the version number of the firmware.

#### Returns

```
const char* return the version number of the firmware
```

Definition at line [949](#) of file [Device.hpp](#).

Referenced by [firmwareVersion\(\)](#).

#### ◆ [getConnectionType\(\)](#)

```
const char * ob::DeviceInfo::getConnectionType ( ) const
```

inline

Get the connection type of the device.

### Returns

```
const char* the connection type of the device, currently supports: "USB", "USB1.0", "USB1.1",
"USB2.0", "USB2.1", "USB3.0", "USB3.1", "USB3.2", "Ethernet"
```

Definition at line [962](#) of file [Device.hpp](#).

Referenced by [connectionType\(\)](#).

## ◆ [getIpAddress\(\)](#)

```
const char * ob::DeviceInfo::getIpAddress ( ) const
```

inline

Get the IP address of the device.

### Attention

Only valid for network devices, otherwise it will return "0.0.0.0".

### Returns

```
const char* the IP address of the device, such as "192.168.1.10"
```

Definition at line [976](#) of file [Device.hpp](#).

Referenced by [ipAddress\(\)](#).

## ◆ [getHardwareVersion\(\)](#)

```
const char * ob::DeviceInfo::getHardwareVersion ( ) const
```

inline

Get the version number of the hardware.

### Returns

```
const char* the version number of the hardware
```

Definition at line [988](#) of file [Device.hpp](#).

Referenced by [hardwareVersion\(\)](#).

## ◆ [getSupportedMinSdkVersion\(\)](#)

```
const char * ob::DeviceInfo::getSupportedMinSdkVersion ( ) const
```

inline

Get the minimum version number of the SDK supported by the device.

**Returns**

const char\* the minimum SDK version number supported by the device

Definition at line **1000** of file [Device.hpp](#).

Referenced by [supportedMinSdkVersion\(\)](#).

◆ [getAsicName\(\)](#)

```
const char * ob::DeviceInfo::getAsicName ( ) const
```

inline

Get chip type name.

**Returns**

const char\* the chip type name

Definition at line **1012** of file [Device.hpp](#).

Referenced by [asicName\(\)](#).

◆ [getDeviceType\(\)](#)

```
OBDeviceType ob::DeviceInfo::getDeviceType ( ) const
```

inline

Get the device type.

**Returns**

[OBDeviceType](#) the device type

Definition at line **1024** of file [Device.hpp](#).

Referenced by [deviceType\(\)](#).

◆ [name\(\)](#)

```
const char * ob::DeviceInfo::name( ) const inline
```

Definition at line [1033](#) of file [Device.hpp](#).

Referenced by [getAsicName\(\)](#), and [getName\(\)](#).

◆ [pid\(\)](#)

```
int ob::DeviceInfo::pid( ) const inline
```

Definition at line [1037](#) of file [Device.hpp](#).

Referenced by [getPid\(\)](#).

◆ [vid\(\)](#)

```
int ob::DeviceInfo::vid( ) const inline
```

Definition at line [1041](#) of file [Device.hpp](#).

Referenced by [getVid\(\)](#).

◆ [uid\(\)](#)

```
const char * ob::DeviceInfo::uid( ) const inline
```

Definition at line [1045](#) of file [Device.hpp](#).

Referenced by [getUid\(\)](#).

◆ [serialNumber\(\)](#)

```
const char * ob::DeviceInfo::serialNumber( ) const inline
```

Definition at line [1049](#) of file [Device.hpp](#).

◆ [firmwareVersion\(\)](#)

```
const char * ob::DeviceInfo::firmwareVersion ( ) const
```

inline

Definition at line **1053** of file [Device.hpp](#).

◆ [connectionType\(\)](#)

```
const char * ob::DeviceInfo::connectionType ( ) const
```

inline

Definition at line **1057** of file [Device.hpp](#).

◆ [ipAddress\(\)](#)

```
const char * ob::DeviceInfo::ipAddress ( ) const
```

inline

Definition at line **1061** of file [Device.hpp](#).

◆ [hardwareVersion\(\)](#)

```
const char * ob::DeviceInfo::hardwareVersion ( ) const
```

inline

Definition at line **1065** of file [Device.hpp](#).

◆ [supportedMinSdkVersion\(\)](#)

```
const char * ob::DeviceInfo::supportedMinSdkVersion ( ) const
```

inline

Definition at line **1069** of file [Device.hpp](#).

◆ [asicName\(\)](#)

```
const char * ob::DeviceInfo::asicName ( ) const
```

inline

Definition at line **1073** of file [Device.hpp](#).

◆ deviceType()

**OBDeviceType** ob::DeviceInfo::deviceType ( ) const

inline

Definition at line **1077** of file **Device.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Device.hpp**

Generated on for OrbbecSDK by **doxygen** 1.14.0

## ob::DeviceList Member List

This is the complete list of members for **ob::DeviceList**, including all inherited members.

<b>connectionType</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>deviceCount</b> () const	<b>ob::DeviceList</b> inline
<b>DeviceList</b> (ob_device_list_t *impl)	<b>ob::DeviceList</b> inline explicit
<b>getConnectionType</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getCount</b> () const	<b>ob::DeviceList</b> inline
<b>getDevice</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getDeviceBySN</b> (const char *serialNumber) const	<b>ob::DeviceList</b> inline
<b>getDeviceByUid</b> (const char *uid) const	<b>ob::DeviceList</b> inline
<b>getIpAddress</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getLocalMacAddress</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getName</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getPid</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getSerialNumber</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getUid</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>getVid</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>ipAddress</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>name</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>pid</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>serialNumber</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>uid</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>vid</b> (uint32_t index) const	<b>ob::DeviceList</b> inline
<b>~DeviceList</b> () noexcept	<b>ob::DeviceList</b> inline

# ob::DeviceList Class Reference

Class representing a list of devices. [More...](#)

```
#include <Device.hpp>
```

## Public Member Functions

```
DeviceList (ob_device_list_t *impl)  
~DeviceList () noexcept  
uint32_t getCount () const  
    Get the number of devices in the list.  
int getPid (uint32_t index) const  
    Get the PID of the device at the specified index.  
int getVid (uint32_t index) const  
    Get the VID of the device at the specified index.  
const char * getUid (uint32_t index) const  
    Get the UID of the device at the specified index.  
const char * getSerialNumber (uint32_t index) const  
    Get the serial number of the device at the specified index.  
const char * getName (uint32_t index) const  
    Get the name of the device at the specified index in the device list.  
const char * getConnectionType (uint32_t index) const  
    Get device connection type.  
const char * getIpAddress (uint32_t index) const  
    get the ip address of the device at the specified index  
const char * getLocalMacAddress (uint32_t index) const  
    get the local mac address of the device at the specified index  
std::shared_ptr< Device > getDevice (uint32_t index) const  
    Get the device object at the specified index.  
std::shared_ptr< Device > getDeviceBySN (const char *serialNumber) const  
    Get the device object with the specified serial number.  
std::shared_ptr< Device > getDeviceByUid (const char *uid) const  
    Get the specified device object from the device list by uid.  
uint32_t deviceCount () const  
    int pid (uint32_t index) const  
    int vid (uint32_t index) const
```

```
const char * uid (uint32_t index) const  
const char * serialNumber (uint32_t index) const  
const char * name (uint32_t index) const  
const char * connectionType (uint32_t index) const  
const char * ipAddress (uint32_t index) const
```

## Detailed Description

Class representing a list of devices.

Definition at line **1085** of file **Device.hpp**.

## Constructor & Destructor Documentation

### ◆ DeviceList()

```
ob::DeviceList::DeviceList ( ob_device_list_t * impl )inline explicit
```

Definition at line **1090** of file **Device.hpp**.

### ◆ ~DeviceList()

```
ob::DeviceList::~DeviceList ( )inline noexcept
```

Definition at line **1091** of file **Device.hpp**.

## Member Function Documentation

### ◆ getCount()

```
uint32_t ob::DeviceList::getCount ( ) const
```

inline

Get the number of devices in the list.

**Returns**

uint32\_t the number of devices in the list

Definition at line [1102](#) of file [Device.hpp](#).

Referenced by [deviceCount\(\)](#).

◆ [getPid\(\)](#)

```
int ob::DeviceList::getPid ( uint32_t index ) const
```

inline

Get the PID of the device at the specified index.

**Parameters**

**index** the index of the device

**Returns**

int the PID of the device

Definition at line [1115](#) of file [Device.hpp](#).

Referenced by [pid\(\)](#).

◆ [getVid\(\)](#)

```
int ob::DeviceList::getVid ( uint32_t index ) const
```

inline

Get the VID of the device at the specified index.

**Parameters**

**index** the index of the device

**Returns**

int the VID of the device

Definition at line [1128](#) of file [Device.hpp](#).

Referenced by [vid\(\)](#).

◆ [getUid\(\)](#)

```
const char * ob::DeviceList::getUid ( uint32_t index ) const
```

inline

Get the UID of the device at the specified index.

**Parameters**

**index** the index of the device

**Returns**

const char\* the UID of the device

Definition at line [1141](#) of file [Device.hpp](#).

Referenced by [uid\(\)](#).

◆ [getSerialNumber\(\)](#)

```
const char * ob::DeviceList::getSerialNumber ( uint32_t index ) const
```

inline

Get the serial number of the device at the specified index.

**Parameters**

**index** the index of the device

**Returns**

const char\* the serial number of the device

Definition at line [1154](#) of file [Device.hpp](#).

Referenced by [serialNumber\(\)](#).

◆ [getName\(\)](#)

```
const char * ob::DeviceList::getName ( uint32_t index ) const
```

inline

Get the name of the device at the specified index in the device list.

This function retrieves the name of the device at the given index in the device list. If an error occurs during the operation, it will be handled by the [Error::handle](#) function.

### Parameters

**index** The index of the device in the device list.

### Returns

const char\* The name of the device at the specified index.

Definition at line [1170](#) of file [Device.hpp](#).

Referenced by [name\(\)](#).

### ◆ [getConnectionType\(\)](#)

```
const char * ob::DeviceList::getConnectionType ( uint32_t index ) const
```

inline

Get device connection type.

### Parameters

**index** device index

### Returns

const char\* returns connection type, currently supports: "USB", "USB1.0", "USB1.1", "USB2.0",  
"USB2.1", "USB3.0", "USB3.1", "USB3.2", "Ethernet"

Definition at line [1183](#) of file [Device.hpp](#).

Referenced by [connectionType\(\)](#).

### ◆ [getIpAddress\(\)](#)

```
const char * ob::DeviceList::getIpAddress ( uint32_t index ) const inline
```

get the ip address of the device at the specified index

### Attention

Only valid for network devices, otherwise it will return "0.0.0.0".

### Parameters

**index** the index of the device

### Returns

const char\* the ip address of the device

Definition at line [1198](#) of file [Device.hpp](#).

Referenced by [ipAddress\(\)](#).

### ◆ [getLocalMacAddress\(\)](#)

```
const char * ob::DeviceList::getLocalMacAddress ( uint32_t index ) const inline
```

get the local mac address of the device at the specified index

### Attention

Only valid for network devices, otherwise it will return "0:0:0:0:0:0".

### Parameters

**index** the index of the device

### Returns

const char\* the local mac address of the device

Definition at line [1213](#) of file [Device.hpp](#).

### ◆ [getDevice\(\)](#)

```
std::shared_ptr<Device> ob::DeviceList::getDevice ( uint32_t index ) const
```

inline

Get the device object at the specified index.

### Attention

If the device has already been acquired and created elsewhere, repeated acquisition will throw an exception

### Parameters

**index** the index of the device to create

### Returns

```
std::shared_ptr<Device> the device object
```

Definition at line [1228](#) of file [Device.hpp](#).

## ◆ [getDeviceBySN\(\)](#)

```
std::shared_ptr<Device> ob::DeviceList::getDeviceBySN ( const char * serialNumber ) const
```

inline

Get the device object with the specified serial number.

### Attention

If the device has already been acquired and created elsewhere, repeated acquisition will throw an exception

### Parameters

**serialNumber** the serial number of the device to create

### Returns

```
std::shared_ptr<Device> the device object
```

Definition at line [1243](#) of file [Device.hpp](#).

## ◆ [getDeviceByUid\(\)](#)

```
std::shared_ptr<Device> ob::DeviceList::getDeviceByUid ( const char * uid ) const inline
```

Get the specified device object from the device list by uid.

On Linux platform, for usb device, the uid of the device is composed of bus-port-dev, for example 1-1.2-1. But the SDK will remove the dev number and only keep the bus-port as the uid to create the device, for example 1-1.2, so that we can create a device connected to the specified USB port. Similarly, users can also directly pass in bus-port as uid to create device.

For GMSL device, the uid is GMSL port with "gmsl2-" prefix, for example gmsl2-1.

### Attention

If the device has been acquired and created elsewhere, repeated acquisition will throw an exception

### Parameters

**uid** The uid of the device to be created

### Returns

std::shared\_ptr<Device> returns the device object

Definition at line [1262](#) of file [Device.hpp](#).

### ◆ deviceCount()

```
uint32_t ob::DeviceList::deviceCount ( ) const inline
```

Definition at line [1271](#) of file [Device.hpp](#).

### ◆ pid()

```
int ob::DeviceList::pid ( uint32_t index ) const inline
```

Definition at line [1275](#) of file [Device.hpp](#).

Referenced by [getPid\(\)](#).

### ◆ vid()

```
int ob::DeviceList::vid ( uint32_t index ) const
```

inline

Definition at line [1279](#) of file [Device.hpp](#).

Referenced by [getVid\(\)](#).

◆ [uid\(\)](#)

```
const char * ob::DeviceList::uid ( uint32_t index ) const
```

inline

Definition at line [1283](#) of file [Device.hpp](#).

Referenced by [getDeviceByUid\(\)](#), and [getUid\(\)](#).

◆ [serialNumber\(\)](#)

```
const char * ob::DeviceList::serialNumber ( uint32_t index ) const
```

inline

Definition at line [1287](#) of file [Device.hpp](#).

Referenced by [getDeviceBySN\(\)](#).

◆ [name\(\)](#)

```
const char * ob::DeviceList::name ( uint32_t index ) const
```

inline

Definition at line [1291](#) of file [Device.hpp](#).

Referenced by [getName\(\)](#).

◆ [connectionType\(\)](#)

```
const char * ob::DeviceList::connectionType ( uint32_t index ) const
```

inline

Definition at line [1295](#) of file [Device.hpp](#).

◆ [ipAddress\(\)](#)

```
const char * ob::DeviceList::ipAddress ( uint32_t index ) const
```

inline

Definition at line **1299** of file [Device.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Device.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::DevicePresetList Member List

This is the complete list of members for **ob::DevicePresetList**, including all inherited members.

<b>count()</b>	<b>ob::DevicePresetList</b> inline
<b>DevicePresetList</b> (ob_device_preset_list_t *impl)	<b>ob::DevicePresetList</b> inline explicit
<b>getCount()</b>	<b>ob::DevicePresetList</b> inline
<b>getName</b> (uint32_t index)	<b>ob::DevicePresetList</b> inline
<b>hasPreset</b> (const char *name)	<b>ob::DevicePresetList</b> inline
<b>~DevicePresetList()</b> noexcept	<b>ob::DevicePresetList</b> inline

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::DevicePresetList Class Reference

Class representing a list of device presets. [More...](#)

```
#include <Device.hpp>
```

## Public Member Functions

**DevicePresetList** (ob\_device\_preset\_list\_t \*impl)

**~DevicePresetList** () noexcept

**uint32\_t getCount** ()

Get the number of device presets in the list.

**const char \* getName** (uint32\_t index)

Get the name of the device preset at the specified index.

**bool hasPreset** (const char \*name)

check if the preset list contains the special name preset.

**uint32\_t count** ()

## Detailed Description

Class representing a list of device presets.

A device preset is a set of parameters or configurations that can be applied to the device to achieve a specific effect or function.

Definition at line [1362](#) of file [Device.hpp](#).

## Constructor & Destructor Documentation

◆ **DevicePresetList()**

ob::DevicePresetList::DevicePresetList ( ob\_device\_preset\_list\_t \* impl )

inline explicit

Definition at line [1367](#) of file [Device.hpp](#).

◆ **~DevicePresetList()**

```
ob::DevicePresetList::~DevicePresetList ( )
```

inline noexcept

Definition at line [1368](#) of file [Device.hpp](#).

## Member Function Documentation

### ◆ getCount()

```
uint32_t ob::DevicePresetList::getCount ( )
```

inline

Get the number of device presets in the list.

#### Returns

`uint32_t` the number of device presets in the list

Definition at line [1379](#) of file [Device.hpp](#).

Referenced by [count\(\)](#).

### ◆ getName()

```
const char * ob::DevicePresetList::getName ( uint32_t index )
```

inline

Get the name of the device preset at the specified index.

#### Parameters

`index` the index of the device preset

#### Returns

`const char*` the name of the device preset

Definition at line [1392](#) of file [Device.hpp](#).

### ◆ hasPreset()

```
bool ob::DevicePresetList::hasPreset ( const char * name )
```

inline

check if the preset list contains the special name preset.

#### Parameters

**name** The name of the preset

#### Returns

bool Returns true if the special name is found in the preset list, otherwise returns false.

Definition at line [1404](#) of file [Device.hpp](#).

#### ◆ count()

```
uint32_t ob::DevicePresetList::count ( )
```

inline

Definition at line [1413](#) of file [Device.hpp](#).

Referenced by [getCount\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Device.hpp](#)

# ob::DisparityTransform Member List

This is the complete list of members for **ob::DisparityTransform**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>DisparityTransform</b> (const std::string &activationKey="")	<b>ob::DisparityTransform</b>	inline
<b>enable</b> (bool enable) const	<b>ob::Filter</b>	inline virtual
<b>Filter()</b> =default	<b>ob::Filter</b>	protected
<b>Filter</b> (ob_filter *impl)	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema()</b> const	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec()</b> const	<b>ob::Filter</b>	inline virtual
<b>getConfigValue</b> (const std::string &configName) const	<b>ob::Filter</b>	inline virtual
<b>getImpl()</b> const	<b>ob::Filter</b>	inline
<b>getName()</b> const	<b>ob::Filter</b>	inline virtual
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init</b> (ob_filter *impl)	<b>ob::Filter</b>	inline protected
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled()</b> const	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process</b> (std::shared_ptr< const Frame > frame) const	<b>ob::Filter</b>	inline virtual
<b>pushFrame</b> (std::shared_ptr< Frame > frame) const	<b>ob::Filter</b>	inline virtual
<b>reset()</b> const	<b>ob::Filter</b>	inline virtual
<b>setCallBack</b> (FilterCallback callback)	<b>ob::Filter</b>	inline virtual
<b>setConfigValue</b> (const std::string &configName, double value) const	<b>ob::Filter</b>	inline virtual
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~DisparityTransform()</b> noexcept override=default	<b>ob::DisparityTransform</b>	
<b>~Filter()</b> noexcept	<b>ob::Filter</b>	inline virtual

# ob::DisparityTransform Class Reference

Depth to disparity or disparity to depth. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::DisparityTransform:

## Public Member Functions

[\*\*DisparityTransform\*\*](#) (const std::string &activationKey="")

[\*\*~DisparityTransform\*\*](#) () noexcept override=default

Public Member Functions inherited from [ob::Filter](#)

## Additional Inherited Members

Protected Member Functions inherited from [ob::Filter](#)

Protected Attributes inherited from [ob::Filter](#)

## Detailed Description

Depth to disparity or disparity to depth.

Definition at line [956](#) of file [Filter.hpp](#).

## Constructor & Destructor Documentation

### ◆ [\*\*DisparityTransform\(\)\*\*](#)

ob::DisparityTransform::DisparityTransform ( const std::string & activationKey = "" )

inline

Definition at line [958](#) of file [Filter.hpp](#).

### ◆ [\*\*~DisparityTransform\(\)\*\*](#)

ob::DisparityTransform::~DisparityTransform ( )

override default noexcept

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::Error Member List

This is the complete list of members for **ob::Error**, including all inherited members.

<b>getArgs()</b> const noexcept	<b>ob::Error</b> inline
<b>getExceptionType()</b> const noexcept	<b>ob::Error</b> inline
<b>getFunction()</b> const noexcept	<b>ob::Error</b> inline
<b>getMessage()</b> const noexcept	<b>ob::Error</b> inline
<b>getName()</b> const noexcept	<b>ob::Error</b> inline
<b>handle</b> (ob_error **error, bool throw_exception=true)	<b>ob::Error</b> inline static
<b>what()</b> const noexcept override	<b>ob::Error</b> inline
<b>~Error()</b> override	<b>ob::Error</b> inline

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::Error Class Reference

```
#include <Error.hpp>
```

Inheritance diagram for ob::Error:

## Public Member Functions

`~Error()` override

Destroy the `Error` object.

`const char * what()` const noexcept override

Returns the error message of the exception.

`OBExceptionType getExceptionType()` const noexcept

Returns the exception type of the exception.

`const char * getFunction()` const noexcept

Returns the name of the function where the exception occurred.

`const char * getArgs()` const noexcept

Returns the arguments of the function where the exception occurred.

`const char * getMessage()` const noexcept

Returns the error message of the exception.

`const char * getName()` const noexcept

## Static Public Member Functions

`static void handle(ob_error **error, bool throw_exception=true)`

A static function to handle the `ob_error` and throw an exception if needed.

## Detailed Description

Definition at line **16** of file [Error.hpp](#).

## Constructor & Destructor Documentation

- ◆ `~Error()`

```
ob::Error::~Error( )
```

inline override

Destroy the **Error** object.

Definition at line 55 of file [Error.hpp](#).

## Member Function Documentation

### ◆ handle()

```
void ob::Error::handle( ob_error ** error,  
                      bool throw_exception = true )
```

inline static

A static function to handle the **ob\_error** and throw an exception if needed.

#### Parameters

**error** The **ob\_error** pointer to be handled.

**throw\_exception** A boolean value to indicate whether to throw an exception or not, the default value is true.

Definition at line 38 of file [Error.hpp](#).

Referenced by [ob::Align::Align\(\)](#), [ob::Frame::as\(\)](#), [ob::Frame::as\(\)](#),  
[ob::StreamProfile::bindExtrinsicTo\(\)](#), [ob::StreamProfile::bindExtrinsicTo\(\)](#),  
[ob::CoordinateTransformHelper::calibration2dTo2d\(\)](#),  
[ob::CoordinateTransformHelper::calibration2dTo3d\(\)](#),  
[ob::CoordinateTransformHelper::calibration3dTo2d\(\)](#),  
[ob::CoordinateTransformHelper::calibration3dTo3d\(\)](#), [ob::Config::Config\(\)](#),  
[ob::Context::Context\(\)](#), [ob::StreamProfileFactory::create\(\)](#), [ob::FilterFactory::createFilter\(\)](#),  
[ob::FrameFactory::createFrame\(\)](#), [ob::FrameFactory::createFrameFromBuffer\(\)](#),  
[ob::FrameFactory::createFrameFromOtherFrame\(\)](#),  
[ob::FrameFactory::createFrameFromStreamProfile\(\)](#), [ob::Context::createNetDevice\(\)](#),  
[ob::FilterFactory::createPrivateFilter\(\)](#), [ob::Sensor::createRecommendedFilters\(\)](#),  
[ob::FrameFactory::createVideoFrame\(\)](#), [ob::FrameFactory::createVideoFrameFromBuffer\(\)](#),  
[ob::DecimationFilter::DecimationFilter\(\)](#), [ob::Config::disableAllStream\(\)](#),  
[ob::Pipeline::disableFrameSync\(\)](#), [ob::Config::disableStream\(\)](#),  
[ob::DisparityTransform::DisparityTransform\(\)](#), [ob::Filter::enable\(\)](#),  
[ob::Config::enableAccelStream\(\)](#), [ob::Config::enableAllStream\(\)](#),  
[ob::Context::enableDeviceClockSync\(\)](#), [ob::Pipeline::enableFrameSync\(\)](#),  
[ob::Device::enableGlobalTimestamp\(\)](#), [ob::Config::enableGyroStream\(\)](#),  
[ob::Device::enableHeartbeat\(\)](#), [ob::Context::enableNetDeviceEnumeration\(\)](#),  
[ob::Config::enableStream\(\)](#), [ob::Config::enableStream\(\)](#), [ob::Config::enableVideoStream\(\)](#),

ob::Device::exportSettingsAsPresetJsonFile(), ob::FormatConvertFilter::FormatConvertFilter(),  
ob::Context::freeIdleMemory(), ob::StreamProfileList::getAccelStreamProfile(),  
ob::DeviceInfo::getAsicName(), ob::Device::getAvailableFrameInterleaveList(),  
ob::Device::getAvailablePresetList(), ob::Device::getAvailablePresetResolutionConfigList(),  
ob::Device::getBoolProperty(), ob::Device::getBoolPropertyRange(),  
ob::Device::getCalibrationCameraParamList(), ob::Pipeline::getCalibrationParam(),  
ob::CameraParamList::getCameraParam(), ob::Pipeline::getCameraParam(),  
ob::Pipeline::getCameraParamWithProfile(), ob::Pipeline::getConfig(),  
ob::Filter::getConfigSchema(), ob::Filter::getConfigValue(),  
ob::DeviceInfo::getConnectionType(), ob::DeviceList::getConnectionType(),  
ob::PointsFrame::getCoordinateValueScale(), ob::CameraParamList::getCount(),  
ob::DeviceFrameInterleaveList::getCount(), ob::DeviceList::getCount(),  
ob::DevicePresetList::getCount(), ob::FrameSet::getCount(),  
ob::OBDepthWorkModeList::getCount(), ob::OBFilterList::getCount(),  
ob::PresetResolutionConfigList::getCount(), ob::SensorList::getCount(),  
ob::StreamProfileList::getCount(), ob::Device::getCurrentDepthModeName(),  
ob::Device::getCurrentDepthWorkMode(), ob::Device::getCurrentPresetName(),  
ob::Pipeline::getD2CDepthProfileList(), ob::Frame::getData(), ob::Frame::getDataSize(),  
ob::Device::getDepthWorkModeList(), ob::DeviceList::getDevice(), ob::Frame::getDevice(),  
ob::Pipeline::getDevice(), ob::DeviceList::getDeviceBySN(), ob::DeviceList::getDeviceByUid(),  
ob::Device::getDeviceInfo(), ob::DeviceInfo::getDeviceType(),  
ob::VideoStreamProfile::getDistortion(), ob::PlaybackDevice::getDuration(),  
ob::Config::getEnabledStreamProfileList(), ob::Device::getExtensionInfo(),  
ob::StreamProfile::getExtrinsicTo(), ob::OBFilterList::getFilter(),  
ob::FilterFactory::getFilterVendorSpecificCode(), ob::DeviceInfo::getFirmwareVersion(),  
ob::Device::getFloatProperty(), ob::Device::getFloatPropertyRange(), ob::Frame::getFormat(),  
ob::StreamProfile::getFormat(), ob::VideoStreamProfile::getFps(), ob::FrameSet::getFrame(),  
ob::FrameSet::getFrameByIndex(), ob::AccelStreamProfile::getFullScaleRange(),  
ob::GyroStreamProfile::getFullScaleRange(), ob::Frame::getGlobalTimeStampUs(),  
ob::StreamProfileList::getGyroStreamProfile(), ob::DeviceInfo::getHardwareVersion(),  
ob::PointsFrame::getHeight(), ob::VideoFrame::getHeight(),  
ob::VideoStreamProfile::getHeight(), ob::Frame::getIndex(), ob::Device::getIntProperty(),  
ob::Device::getIntPropertyRange(), ob::AccelStreamProfile::getIntrinsic(),  
ob::GyroStreamProfile::getIntrinsic(), ob::VideoStreamProfile::getIntrinsic(),  
ob::DeviceInfo::getIpAddress(), ob::DeviceList::getIpAddress(),  
ob::DeviceList::getLocalMacAddress(), ob::Frame::getMetadata(), ob::Frame::getMetadataSize(),  
ob::Frame::getMetadataValue(), ob::Device::getMultiDeviceSyncConfig(),  
ob::DeviceFrameInterleaveList::getName(), ob::DeviceInfo::getName(),  
ob::DeviceList::getName(), ob::DevicePresetList::getName(),  
ob::OBDepthWorkModeList::getOBDepthWorkMode(), ob::DeviceInfo::getPid(),  
ob::DeviceList::getPid(), ob::VideoFrame::getPixelAvailableBitSize(),  
ob::VideoFrame::getPixelType(), ob::PlaybackDevice::getPlaybackStatus(),  
ob::PlaybackDevice::getPosition(),  
ob::PresetResolutionConfigList::getPresetResolutionRatioConfig(),

ob::StreamProfileList::getProfile(), ob::AccelStreamProfile::getSampleRate(),  
ob::GyroStreamProfile::getSampleRate(), ob::Device::getSensor(), ob::Frame::getSensor(),  
ob::SensorList::getSensor(), ob::SensorList::getSensor(), ob::Device::getSensorList(),  
ob::SensorList::getSensorType(), ob::DeviceInfo::getSerialNumber(),  
ob::DeviceList::getSerialNumber(), ob::Frame::getStreamProfile(),  
ob::Pipeline::getStreamProfileList(), ob::Sensor::getStreamProfileList(),  
ob::Device::getStructuredData(), ob::DeviceInfo::getSupportedMinSdkVersion(),  
ob::Device::getSupportedMultiDeviceSyncModeBitmap(), ob::Device::getSupportedProperty(),  
ob::Device::getSupportedPropertyCount(), ob::Frame::getSystemTimeStampUs(),  
ob::AccelFrame::getTemperature(), ob::GyroFrame::getTemperature(),  
ob::Device::getTimestampResetConfig(), ob::Frame::getTimeStampUs(), ob::Frame::getType(),  
ob::Sensor::getType(), ob::StreamProfile::getType(), ob::DeviceInfo::getUid(),  
ob::DeviceList::getUid(), ob::AccelFrame::getValue(), ob::GyroFrame::getValue(),  
ob::DepthFrame::getValueScale(), ob::DeviceInfo::getVid(), ob::DeviceList::getVid(),  
ob::StreamProfileList::getVideoStreamProfile(), ob::PointsFrame::getWidth(),  
ob::VideoFrame::getWidth(), ob::VideoStreamProfile::getWidth(),  
ob::DeviceFrameInterleaveList::hasFrameInterleave(), ob::Frame::hasMetadata(),  
ob::DevicePresetList::hasPreset(), ob::HdrMerge::HdrMerge(),  
ob::HoleFillingFilter::HoleFillingFilter(), ob::Filter::init(), ob::Filter::isEnabled(),  
ob::Device::isExtensionInfoExist(), ob::Device::isFrameInterleaveSupported(),  
ob::Device::isGlobalTimestampSupported(), ob::Device::isPropertySupported(),  
ob::Device::loadFrameInterleave(), ob::Device::loadPreset(),  
ob::Device::loadPresetFromFile(), ob::NoiseRemovalFilter::NoiseRemovalFilter(),  
ob::Device::operator=(), ob::Sensor::operator=(), ob::StreamProfile::operator=(),  
ob::PlaybackDevice::pause(), ob::RecordDevice::pause(), ob::Pipeline::Pipeline(),  
ob::Pipeline::Pipeline(), ob::PlaybackDevice::PlaybackDevice(),  
ob::PointCloudFilter::PointCloudFilter(), ob::Filter::process(), ob::Filter::pushFrame(),  
ob::FrameSet::pushFrame(), ob::Context::queryDeviceList(), ob::Device::readCustomerData(),  
ob::Device::reboot(), ob::RecordDevice::RecordDevice(), ob::Filter::reset(),  
ob::PlaybackDevice::resume(), ob::RecordDevice::resume(),  
ob::PointCloudHelper::savePointCloudToPly(), ob::PlaybackDevice::seek(),  
ob::Device::sendAndReceiveData(), ob::SequenceIdFilter::SequenceIdFilter(),  
ob::Config::setAlignMode(), ob::Align::setAlignToStreamProfile(), ob::Device::setBoolProperty(),  
ob::Filter::setCallBack(), ob::Filter::setConfigValue(), ob::Config::setDepthScaleRequire(),  
ob::Context::setDeviceChangedCallback(), ob::Device::setDeviceStateChangedCallback(),  
ob::VideoStreamProfile::setDistortion(), ob::Context::setExtensionsDirectory(),  
ob::Device::setFloatProperty(), ob::Config:: setFrameAggregateOutputMode(),  
ob::FrameHelper::setFrameDeviceTimestampUs(), ob::Device::setIntProperty(),  
ob::VideoStreamProfile::setIntrinsic(), ob::Context::setLoggerSeverity(),  
ob::Context::setLoggerToCallback(), ob::Context::setLoggerToConsole(),  
ob::Context::setLoggerToFile(), ob::Device::setMultiDeviceSyncConfig(),  
ob::PlaybackDevice::setPlaybackRate(),  
ob::PlaybackDevice::setPlaybackStatusChangeCallback(), ob::Device::setStructuredData(),  
ob::Device::setTimestampResetConfig(), ob::Context::setUvcBackendType(),

`ob::SpatialAdvancedFilter::SpatialAdvancedFilter()`, `ob::Pipeline::start()`, `ob::Pipeline::start()`,  
`ob::Sensor::start()`, `ob::Pipeline::stop()`, `ob::Sensor::stop()`, `ob::Device::switchDepthWorkMode()`,  
`ob::Device::switchDepthWorkMode()`, `ob::Sensor::switchProfile()`,  
`ob::TemporalFilter::TemporalFilter()`, `ob::ThresholdFilter::ThresholdFilter()`,  
`ob::Device::timerSyncWithHost()`, `ob::Device::timestampReset()`,  
`ob::CoordinateTransformHelper::transformation2dto2d()`,  
`ob::CoordinateTransformHelper::transformation2dto3d()`,  
`ob::CoordinateTransformHelper::transformation3dto2d()`,  
`ob::CoordinateTransformHelper::transformation3dto3d()`,  
`ob::CoordinateTransformHelper::transformationDepthFrameToColorCamera()`,  
`ob::CoordinateTransformHelper::transformationDepthToPointCloud()`,  
`ob::CoordinateTransformHelper::transformationDepthToRGBDPointCloud()`,  
`ob::CoordinateTransformHelper::transformationInitXYTables()`, `ob::Device::triggerCapture()`,  
`ob::Device::updateFirmware()`, `ob::Device::updateFirmwareFromData()`,  
`ob::Device::updateOptionalDepthPresets()`, `ob::Pipeline::waitForFrameset()`,  
`ob::Device::writeCustomerData()`, `ob::CameraParamList::~CameraParamList()`,  
`ob::Config::~Config()`, `ob::Context::~Context()`, `ob::Device::~Device()`,  
`ob::DeviceFrameInterleaveList::~DeviceFrameInterleaveList()`, `ob::DeviceInfo::~DeviceInfo()`,  
`ob::DeviceList::~DeviceList()`, `ob::DevicePresetList::~DevicePresetList()`, `ob::Filter::~Filter()`,  
`ob::Frame::~Frame()`, `ob::OBDepthWorkModeList::~OBDepthWorkModeList()`,  
`ob::OBFilterList::~OBFilterList()`, `ob::Pipeline::~Pipeline()`,  
`ob::PresetResolutionConfigList::~PresetResolutionConfigList()`,  
`ob::RecordDevice::~RecordDevice()`, `ob::Sensor::~Sensor()`, `ob::SensorList::~SensorList()`,  
`ob::StreamProfile::~StreamProfile()`, and `ob::StreamProfileList::~StreamProfileList()`.

◆ `what()`

```
const char * ob::Error::what ( ) const
```

inline override noexcept

Returns the error message of the exception.

**Returns**

`const char*` The error message.

Definition at line [67](#) of file [Error.hpp](#).

◆ `getExceptionType()`

**OBExceptionType** ob::Error::getExceptionType ( ) const

inline noexcept

Returns the exception type of the exception.

Read the comments of the **OBExceptionType** enum in the **libobsensor/h/ObTypes.h** file for more information.

## Returns

**OBExceptionType** The exception type.

Definition at line 77 of file **Error.hpp**.

#### ◆ getFunction()

```
const char * ob::Error::getFunction ( ) const
```

inline noexcept

Returns the name of the function where the exception occurred.

## Returns

`const char*` The function name.

Definition at line **86** of file **Error.hpp**.

## ◆ getArgs()

```
const char * ob::Error::getArgs ( ) const
```

inline noexcept

Returns the arguments of the function where the exception occurred.

## Returns

const char\* The arguments.

Definition at line **95** of file **Error.hpp**.

#### ◆ getMessage()

```
const char * ob::Error::getMessage( ) const           inline noexcept
```

Returns the error message of the exception.

It is recommended to use the [what\(\)](#) function instead.

#### Returns

const char\* The error message.

Definition at line [105](#) of file [Error.hpp](#).

#### ◆ [getName\(\)](#)

```
const char * ob::Error::getName( ) const           inline noexcept
```

Definition at line [111](#) of file [Error.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Error.hpp](#)

# ob::Filter Member List

This is the complete list of members for **ob::Filter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b> inline
<b>callback_</b>	<b>ob::Filter</b> protected
<b>configSchemaVec_</b>	<b>ob::Filter</b> protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b> inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b> protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b> inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b> inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b> inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b> inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b> inline
<b>getName() const</b>	<b>ob::Filter</b> inline virtual
<b>impl_</b>	<b>ob::Filter</b> protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b> inline protected virtual
<b>is()</b>	<b>ob::Filter</b>
<b>isEnabled() const</b>	<b>ob::Filter</b> inline virtual
<b>name_</b>	<b>ob::Filter</b> protected
<b>process(std::shared_ptr&lt; const Frame &gt; frame) const</b>	<b>ob::Filter</b> inline virtual
<b>pushFrame(std::shared_ptr&lt; Frame &gt; frame) const</b>	<b>ob::Filter</b> inline virtual
<b>reset() const</b>	<b>ob::Filter</b> inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b> inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b> inline virtual
<b>type()</b>	<b>ob::Filter</b> inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b> inline virtual

# ob::Filter Class Reference

The **Filter** class is the base class for all filters in the SDK. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::Filter:

## Public Member Functions

```
    Filter (ob_filter *impl)  
        virtual ~Filter () noexcept  
        ob_filter * getImpl () const  
            Get the Impl object of the filter.  
  
        virtual const std::string & getName () const  
            Get the type of filter.  
  
        virtual void reset () const  
            Reset the filter, freeing the internal cache, stopping  
            the processing thread, and clearing the pending  
            buffer frame when asynchronous processing is  
            used.  
  
        virtual void enable (bool enable) const  
            enable the filter  
  
        virtual bool isEnabled () const  
            Return Enable State.  
  
        virtual std::shared_ptr< Frame > process (std::shared_ptr< const Frame > frame)  
            const  
            Processes a frame synchronously.  
  
        virtual void pushFrame (std::shared_ptr< Frame > frame) const  
            Pushes the pending frame into the cache for  
            asynchronous processing.  
  
        virtual void setCallBack (FilterCallback callback)  
            Set the callback function for asynchronous  
            processing.  
  
        virtual std::string getConfigSchema () const  
            Get config schema of the filter.  
  
    virtual std::vector< OBFfilterConfigSchemaItem > getConfigSchemaVec () const  
        Get the Config Schema Vec object.  
  
    virtual void setConfigValue (const std::string &configName,  
        double value) const
```

```

        Set the filter config value by name.

    virtual double getConfigValue (const std::string &configName)
        const
            Get the Config Value object by name.

    virtual const char * type ()

template<typename T>
    bool is ()
        Check if the runtime type of the filter object is
        compatible with a given type.

template<typename T>
    std::shared_ptr< T > as ()

```

## Protected Member Functions

```

Filter ()=default
    Default constructor with nullptr impl, used for derived classes only.

virtual void init (ob_filter *impl)

```

## Protected Attributes

```

ob_filter * impl_ = nullptr
    std::string name_
    FilterCallback callback_
    std::vector< OBFilterConfigSchemaItem > configSchemaVec_

```

## Detailed Description

The **Filter** class is the base class for all filters in the SDK.

Definition at line **76** of file **Filter.hpp**.

## Constructor & Destructor Documentation

### ◆ Filter() [1/2]

```
ob::Filter::Filter( )
protected default
```

Default constructor with nullptr impl, used for derived classes only.

◆ Filter() [2/2]

ob::Filter::Filter ( **ob\_filter** \* impl ) inline explicit

Definition at line 112 of file **Filter.hpp**.

◆ ~Filter()

virtual ob::Filter::~Filter ( ) inline virtual noexcept

Definition at line 116 of file **Filter.hpp**.

## Member Function Documentation

◆ init()

virtual void ob::Filter::init ( **ob\_filter** \* impl ) inline protected virtual

Definition at line 89 of file **Filter.hpp**.

Referenced by **ob::Align::Align()**, **ob::DecimationFilter::DecimationFilter()**,  
**ob::DisparityTransform::DisparityTransform()**, **Filter()**,  
**ob::FormatConvertFilter::FormatConvertFilter()**, **ob::HdrMerge::HdrMerge()**,  
**ob::HoleFillingFilter::HoleFillingFilter()**, **ob::NoiseRemovalFilter::NoiseRemovalFilter()**,  
**ob::PointCloudFilter::PointCloudFilter()**, **ob::SequenceIdFilter::SequenceIdFilter()**,  
**ob::SpatialAdvancedFilter::SpatialAdvancedFilter()**, **ob::TemporalFilter::TemporalFilter()**, and  
**ob::ThresholdFilter::ThresholdFilter()**.

◆ getImpl()

**ob\_filter** \* ob::Filter::getImpl ( ) const inline

Get the Impl object of the filter.

### Returns

**ob\_filter\*** The Impl object of the filter.

Definition at line 129 of file **Filter.hpp**.

◆ **getName()**

virtual const std::string & ob::Filter::getName ( ) const

inline virtual

Get the type of filter.

**Returns**

string The type of filte.

Definition at line **138** of file [Filter.hpp](#).

Referenced by [type\(\)](#).

◆ **reset()**

virtual void ob::Filter::reset ( ) const

inline virtual

Reset the filter, freeing the internal cache, stopping the processing thread, and clearing the pending buffer frame when asynchronous processing is used.

Definition at line **146** of file [Filter.hpp](#).

◆ **enable()**

virtual void ob::Filter::enable ( bool enable ) const

inline virtual

enable the filter

Definition at line **155** of file [Filter.hpp](#).

Referenced by [enable\(\)](#), and [isEnabled\(\)](#).

◆ **isEnabled()**

virtual bool ob::Filter::isEnabled ( ) const

inline virtual

Return Enable State.

Definition at line **164** of file [Filter.hpp](#).

◆ **process()**

```
virtual std::shared_ptr<Frame> ob::Filter::process ( std::shared_ptr<const Frame> frame ) const inline virtual
```

Processes a frame synchronously.

#### Parameters

**frame** The frame to be processed.

#### Returns

`std::shared_ptr<Frame>` The processed frame.

Definition at line [177](#) of file [Filter.hpp](#).

### ◆ pushFrame()

```
virtual void ob::Filter::pushFrame ( std::shared_ptr<Frame> frame ) const inline virtual
```

Pushes the pending frame into the cache for asynchronous processing.

#### Parameters

**frame** The pending frame. The processing result is returned by the callback function.

Definition at line [192](#) of file [Filter.hpp](#).

### ◆ setCallBack()

```
virtual void ob::Filter::setCallBack ( FilterCallback callback ) inline virtual
```

Set the callback function for asynchronous processing.

#### Parameters

**callback** The processing result callback.

Definition at line [203](#) of file [Filter.hpp](#).

### ◆ getConfigSchema()

```
virtual std::string ob::Filter::getConfigSchema( ) const           inline virtual
```

Get config schema of the filter.

The returned string is a csv format string representing the configuration schema of the filter. The format of the string is: <parameter\_name>, <parameter\_type: "int", "float", "bool">, <minimum\_value>, <maximum\_value>, <value\_step>, <default\_value>, <parameter\_description>

### Returns

std::string The config schema of the filter.

Definition at line [217](#) of file [Filter.hpp](#).

### ◆ getConfigSchemaVec()

```
virtual std::vector<OBFilterConfigSchemaItem> ob::Filter::getConfigSchemaVec( ) const           inline virtual
```

Get the [Config](#) Schema Vec object.

The returned vector contains the config schema items. Each item in the vector is an [OBFilterConfigSchemaItem](#) object.

### Returns

std::vector<OBFilterConfigSchemaItem> The vector of the filter config schema.

Definition at line [230](#) of file [Filter.hpp](#).

Referenced by [ob::SpatialAdvancedFilter::getAlphaRange\(\)](#),  
[ob::TemporalFilter::getDiffScaleRange\(\)](#), [ob::NoiseRemovalFilter::getDispDiffRange\(\)](#),  
[ob::SpatialAdvancedFilter::getDispDiffRange\(\)](#), [ob::SpatialAdvancedFilter::getMagnitudeRange\(\)](#),  
[ob::ThresholdFilter::getMaxRange\(\)](#), [ob::NoiseRemovalFilter::getMaxSizeRange\(\)](#),  
[ob::ThresholdFilter::getMinRange\(\)](#), [ob::SpatialAdvancedFilter::getRadiusRange\(\)](#), and  
[ob::TemporalFilter::getWeightRange\(\)](#).

### ◆ setConfigValue()

```
virtual void ob::Filter::setConfigValue ( const std::string & configName,  
                                         double           value ) const  
                                         inline virtual
```

Set the filter config value by name.

### Attention

The pass into value type is double, which will be cast to the actual type inside the filter. The actual type can be queried by the filter config schema returned by [getConfigSchemaVec](#).

### Parameters

**configName** The name of the config.  
**value** The value of the config.

Definition at line [243](#) of file [Filter.hpp](#).

Referenced by [ob::Align::Align\(\)](#), [ob::SequenceIdFilter::selectSequenceId\(\)](#),  
[ob::PointCloudFilter::setColorDataNormalization\(\)](#),  
[ob::PointCloudFilter::setCoordinateDataScaled\(\)](#), [ob::PointCloudFilter::setCoordinateSystem\(\)](#),  
[ob::PointCloudFilter::setCreatePointFormat\(\)](#), [ob::TemporalFilter::setDiffScale\(\)](#),  
[ob::HoleFillingFilter::setFilterMode\(\)](#), [ob::NoiseRemovalFilter::setFilterParams\(\)](#),  
[ob::SpatialAdvancedFilter::setFilterParams\(\)](#), [ob::FormatConvertFilter::setFormatConvertType\(\)](#),  
[ob::Align::setMatchTargetResolution\(\)](#), [ob::DecimationFilter::setScaleValue\(\)](#),  
[ob::ThresholdFilter::setValueRange\(\)](#), and [ob::TemporalFilter::setWeight\(\)](#).

◆ [getConfigValue\(\)](#)

```
virtual double ob::Filter::getConfigValue ( const std::string & configName ) const
```

inline virtual

Get the **Config** Value object by name.

### Attention

The returned value type has been casted to double inside the filter. The actual type can be queried by the filter config schema returned by [getConfigSchemaVec](#).

### Parameters

**configName** The name of the config.

### Returns

double The value of the config.

Definition at line [258](#) of file [Filter.hpp](#).

Referenced by [ob::Align::getAlignToStreamType\(\)](#), [ob::SpatialAdvancedFilter::getAlphaRange\(\)](#), [ob::TemporalFilter::getDiffScaleRange\(\)](#), [ob::NoiseRemovalFilter::getDispDiffRange\(\)](#), [ob::SpatialAdvancedFilter::getDispDiffRange\(\)](#), [ob::HoleFillingFilter::getFilterMode\(\)](#), [ob::NoiseRemovalFilter::getFilterParams\(\)](#), [ob::SpatialAdvancedFilter::getFilterParams\(\)](#), [ob::SpatialAdvancedFilter::getMagnitudeRange\(\)](#), [ob::ThresholdFilter::getMaxRange\(\)](#), [ob::NoiseRemovalFilter::getMaxSizeRange\(\)](#), [ob::ThresholdFilter::getMinRange\(\)](#), [ob::SpatialAdvancedFilter::getRadiusRange\(\)](#), [ob::DecimationFilter::getScaleRange\(\)](#), [ob::DecimationFilter::getScaleValue\(\)](#), [ob::SequenceIdFilter::getSelectSequenceId\(\)](#), and [ob::TemporalFilter::getWeightRange\(\)](#).

### ◆ type()

```
virtual const char * ob::Filter::type ( )
```

inline virtual

Definition at line [273](#) of file [Filter.hpp](#).

Referenced by [is\(\)](#), [ob::PointCloudFilter::setCoordinateSystem\(\)](#), and [ob::FormatConvertFilter::setFormatConvertType\(\)](#).

### ◆ is()

```
template<typename T>
bool ob::Filter::is( )
```

Check if the runtime type of the filter object is compatible with a given type.

Define the **is()** template function for the **Filter** class.

### Template Parameters

**T**The given type.

### Returns

bool The result.

Definition at line **1028** of file **Filter.hpp**.

Referenced by **as()**, and **is()**.

### ◆ as()

```
template<typename T>
std::shared_ptr<T> ob::Filter::as( )
```

inline

Definition at line **285** of file **Filter.hpp**.

## Member Data Documentation

### ◆ impl\_

```
ob_filter* ob::Filter::impl_ = nullptr
```

protected

Definition at line **78** of file **Filter.hpp**.

Referenced by **enable()**, **getConfigSchema()**, **getConfigValue()**, **getImpl()**, **init()**, **isEnabled()**, **process()**, **pushFrame()**, **reset()**, **ob::Align::setAlignToStreamProfile()**, **setCallBack()**, **setConfigValue()**, and **~Filter()**.

### ◆ name\_

std::string ob::Filter::name\_

protected

Definition at line **79** of file [Filter.hpp](#).

Referenced by [getName\(\)](#), and [init\(\)](#).

◆ **callback\_**

[FilterCallback](#) ob::Filter::callback\_

protected

Definition at line **80** of file [Filter.hpp](#).

Referenced by [setCallBack\(\)](#).

◆ **configSchemaVec\_**

std::vector<[OBFilterConfigSchemaItem](#)> ob::Filter::configSchemaVec\_

protected

Definition at line **81** of file [Filter.hpp](#).

Referenced by [getConfigSchemaVec\(\)](#), [ob::DecimationFilter::getScaleRange\(\)](#), and [init\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::FilterFactory Member List

This is the complete list of members for **ob::FilterFactory**, including all inherited members.

<b>createFilter</b> (const std::string &name)	<b>ob::FilterFactory</b>	inline static
<b>createPrivateFilter</b> (const std::string &name, const std::string &activationKey)	<b>ob::FilterFactory</b>	inline static
<b>getFilterVendorSpecificCode</b> (const std::string &name)	<b>ob::FilterFactory</b>	inline static

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::FilterFactory Class Reference

A factory class for creating filters. [More...](#)

```
#include <Filter.hpp>
```

## Static Public Member Functions

```
static std::shared_ptr<Filter> createFilter (const std::string &name)
```

Create a filter by name.

```
static std::shared_ptr<Filter> createPrivateFilter (const std::string &name, const std::string  
&activationKey)
```

Create a private filter by name and activation key.

```
static std::string getFilterVendorSpecificCode (const std::string &name)
```

Get the vendor specific code of a filter by filter name.

## Detailed Description

A factory class for creating filters.

Definition at line [297](#) of file [Filter.hpp](#).

## Member Function Documentation

### ◆ **createFilter()**

```
std::shared_ptr<Filter> ob::FilterFactory::createFilter ( const std::string & name )
```

inline static

Create a filter by name.

Definition at line [302](#) of file [Filter.hpp](#).

### ◆ **createPrivateFilter()**

```
std::shared_ptr<Filter> ob::FilterFactory::createPrivateFilter ( const std::string & name,  
const std::string & activationKey ) inline static
```

Create a private filter by name and activation key.

Some private filters require an activation key to be activated, its depends on the vendor of the filter.

### Parameters

**name** The name of the filter.

**activationKey** The activation key of the filter.

Definition at line [316](#) of file [Filter.hpp](#).

## ◆ **getFilterVendorSpecificCode()**

```
std::string ob::FilterFactory::getFilterVendorSpecificCode ( const std::string & name ) inline static
```

Get the vendor specific code of a filter by filter name.

A private filter can define its own vendor specific code for specific purposes.

### Parameters

**name** The name of the filter.

### Returns

`std::string` The vendor specific code of the filter.

Definition at line [330](#) of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::FormatConvertFilter Member List

This is the complete list of members for **ob::FormatConvertFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>FormatConvertFilter()</b>	<b>ob::FormatConvertFilter</b>	inline
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt; const Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt; Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setFormatConvertType(OBConvertFormat type)</b>	<b>ob::FormatConvertFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~FormatConvertFilter() noexcept override=default</b>	<b>ob::FormatConvertFilter</b>	virtual

# ob::FormatConvertFilter Class Reference

The **FormatConvertFilter** class is a subclass of **Filter** that performs format conversion. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::FormatConvertFilter:

## Public Member Functions

**FormatConvertFilter ()**

virtual **~FormatConvertFilter ()** noexcept override=default

void **setFormatConvertType (OBConvertFormat type)**

Set the format conversion type.

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

The **FormatConvertFilter** class is a subclass of **Filter** that performs format conversion.

Definition at line **460** of file **Filter.hpp**.

## Constructor & Destructor Documentation

◆ **FormatConvertFilter()**

ob::FormatConvertFilter::FormatConvertFilter ( )

inline

Definition at line **462** of file **Filter.hpp**.

◆ **~FormatConvertFilter()**

virtual ob::FormatConvertFilter::~FormatConvertFilter ( )

override virtual default noexcept

# Member Function Documentation

## ◆ setFormatConvertType()

void ob::FormatConvertFilter::setFormatConvertType ( **OBConvertFormat** type )

inline

Set the format conversion type.

### Parameters

**type** The format conversion type.

Definition at line **476** of file [Filter.hpp](#).

Referenced by [setFormatConvertType\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::Frame Member List

This is the complete list of members for **ob::Frame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	
<b>metadata() const</b>	<b>ob::Frame</b>	inline
<b>metadataSize() const</b>	<b>ob::Frame</b>	inline
<b>systemTimeStamp() const</b>	<b>ob::Frame</b>	inline
<b>systemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>timeStamp() const</b>	<b>ob::Frame</b>	inline
<b>timeStampUs() const</b>	<b>ob::Frame</b>	inline

**type()** const  
**~Frame()** noexcept

**ob::Frame** inline  
**ob::Frame** inline virtual

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::Frame Class Reference

Define the frame class, which is the base class of all frame types. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::Frame:

## Public Member Functions

**Frame** (const **ob\_frame** \*impl)

Construct a new **Frame** object with a given pointer to the internal frame object.

const **ob\_frame** \* **getImpl** () const

Get the internal (impl) frame object.

virtual **~Frame** () noexcept

Destroy the **Frame** object.

virtual **OBFrameType** **getType** () const

Get the type of frame.

virtual **OBFormat** **getFormat** () const

Get the format of the frame.

virtual uint64\_t **getIndex** () const

Get the sequence number of the frame.

virtual uint8\_t \* **getData** () const

Get frame data.

virtual uint32\_t **getDataSize** () const

Get the size of the frame data.

uint64\_t **getTimeStampUs** () const

Get the hardware timestamp of the frame in microseconds.

uint64\_t **getSystemTimeStampUs** () const

Get the system timestamp of the frame in microseconds.

uint64\_t **getGlobalTimeStampUs** () const

Get the global timestamp of the frame in microseconds.

uint8\_t \* **getMetadata** () const

Get the metadata pointer of the frame.

uint32\_t **getMetadataSize** () const

Get the size of the metadata of the frame.

bool **hasMetadata** (**OBFrameMetadataType** type) const

Check if the frame object has metadata of a given type.

```

int64_t getMetadataValue (OBFrameMetadataType type) const
    Get the metadata value.

std::shared_ptr<StreamProfile> getStreamProfile () const
    get StreamProfile of the frame

std::shared_ptr<Sensor> getSensor () const
    get owner sensor of the frame

std::shared_ptr<Device> getDevice () const
    get owner device of the frame

template<typename T>
    bool is () const
        Check if the runtime type of the frame object is compatible with a
        given type.

template<typename T>
    std::shared_ptr< T > as ()
        Convert the frame object to a target type.

template<typename T>
    std::shared_ptr< const T > as () const
        Convert the frame object to a target type.

OBFrameType type () const
virtual OBFormat format () const
    virtual uint64_t index () const
    virtual void * data () const
    virtual uint32_t dataSize () const
        uint64_t timeStamp () const
        uint64_t timeStampUs () const
        uint64_t systemTimeStamp () const
        uint64_t systemTimeStampUs () const
        uint64_t globalTimeStampUs () const
        uint8_t * metadata () const
        uint32_t metadataSize () const

```

## Protected Attributes

const **ob\_frame** \* **impl\_** = nullptr

The pointer to the internal (c api level) frame object.

## Detailed Description

Define the frame class, which is the base class of all frame types.

Definition at line [45](#) of file [Frame.hpp](#).

## Constructor & Destructor Documentation

### ◆ Frame()

ob::Frame::Frame ( const **ob\_frame** \* impl )

inline explicit

Construct a new **Frame** object with a given pointer to the internal frame object.

#### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

#### Parameters

**impl** The pointer to the internal frame object.

Definition at line [62](#) of file [Frame.hpp](#).

Referenced by [ob::AccelFrame::AccelFrame\(\)](#), [ob::FrameSet::FrameSet\(\)](#),  
[ob::GyroFrame::GyroFrame\(\)](#), [ob::PointsFrame::PointsFrame\(\)](#), and  
[ob::VideoFrame::VideoFrame\(\)](#).

### ◆ ~Frame()

virtual ob::Frame::~Frame ( )

inline virtual noexcept

Destroy the **Frame** object.

Definition at line [76](#) of file [Frame.hpp](#).

## Member Function Documentation

### ◆ getImpl()

```
const ob_frame* ob::Frame::getImpl( ) const inline
```

Get the internal (impl) frame object.

#### Returns

const **ob\_frame**\* the pointer to the internal frame object.

Definition at line **69** of file [Frame.hpp](#).

#### ◆ **getType()**

```
virtual OBFrameType ob::Frame::getType( ) const inline virtual
```

Get the type of frame.

#### Returns

**OBFrameType** The type of frame.

Definition at line **90** of file [Frame.hpp](#).

Referenced by [is\(\)](#), and [type\(\)](#).

#### ◆ **getFormat()**

```
virtual OBFormat ob::Frame::getFormat( ) const inline virtual
```

Get the format of the frame.

#### Returns

**OBFormat** The format of the frame.

Definition at line **103** of file [Frame.hpp](#).

Referenced by [format\(\)](#).

#### ◆ **getIndex()**

```
virtual uint64_t ob::Frame::getIndex( ) const
```

inline virtual

Get the sequence number of the frame.

#### Note

The sequence number for each frame is managed by the SDK. It increments by 1 for each frame on each stream.

#### Returns

uint64\_t The sequence number of the frame.

Definition at line [118](#) of file [Frame.hpp](#).

Referenced by [index\(\)](#).

### ◆ [getData\(\)](#)

```
virtual uint8_t * ob::Frame::getData( ) const
```

inline virtual

Get frame data.

#### Returns

const uint8\_t \* The frame data pointer.

Definition at line [131](#) of file [Frame.hpp](#).

Referenced by [data\(\)](#).

### ◆ [getDataSize\(\)](#)

```
virtual uint32_t ob::Frame::getDataSize( ) const
```

inline virtual

Get the size of the frame data.

#### Returns

uint32\_t The size of the frame data. For point cloud data, this returns the number of bytes occupied by all point sets. To find the number of points, divide the dataSize by the structure size of the corresponding point type.

Definition at line [146](#) of file [Frame.hpp](#).

Referenced by [dataSize\(\)](#).

◆ [getTimeStampUs\(\)](#)

`uint64_t ob::Frame::getTimeStampUs( ) const`

inline

Get the hardware timestamp of the frame in microseconds.

The hardware timestamp is the time point when the frame was captured by the device, on device clock domain.

**Returns**

`uint64_t` The hardware timestamp of the frame in microseconds.

Definition at line [160](#) of file [Frame.hpp](#).

Referenced by [timeStamp\(\)](#), and [timeStampUs\(\)](#).

◆ [getSystemTimeStampUs\(\)](#)

`uint64_t ob::Frame::getSystemTimeStampUs( ) const`

inline

Get the system timestamp of the frame in microseconds.

The system timestamp is the time point when the frame was received by the host, on host clock domain.

**Returns**

`uint64_t` The system timestamp of the frame in microseconds.

Definition at line [174](#) of file [Frame.hpp](#).

Referenced by [systemTimeStamp\(\)](#), and [systemTimeStampUs\(\)](#).

◆ [getGlobalTimeStampUs\(\)](#)

```
uint64_t ob::Frame::getGlobalTimeStampUs ( ) const
```

inline

Get the global timestamp of the frame in microseconds.

The global timestamp is the time point when the frame was captured by the device, and has been converted to the host clock domain. The conversion process base on the device timestamp and can eliminate the timer drift of the device

### Attention

The global timestamp disable by default. If global timestamp is not enabled, the function will return 0. To enable the global timestamp, please call [Device::enableGlobalTimestamp\(\)](#) function.

Only some devices support getting the global timestamp. Check the device support status by [Device::isGlobalTimestampSupported\(\)](#) function.

### Returns

uint64\_t The global timestamp of the frame in microseconds.

Definition at line [193](#) of file [Frame.hpp](#).

Referenced by [globalTimeStampUs\(\)](#).

### ◆ [getMetadata\(\)](#)

```
uint8_t * ob::Frame::getMetadata ( ) const
```

inline

Get the metadata pointer of the frame.

### Returns

const uint8\_t \* The metadata pointer of the frame.

Definition at line [206](#) of file [Frame.hpp](#).

Referenced by [metadata\(\)](#).

### ◆ [getMetadataSize\(\)](#)

```
uint32_t ob::Frame::getMetadataSize( ) const
```

inline

Get the size of the metadata of the frame.

#### Returns

uint32\_t The size of the metadata of the frame.

Definition at line [219](#) of file [Frame.hpp](#).

Referenced by [metadataSize\(\)](#).

#### ◆ [hasMetadata\(\)](#)

```
bool ob::Frame::hasMetadata( OBFrameMetadataType type ) const
```

inline

Check if the frame object has metadata of a given type.

#### Parameters

**type** The metadata type. refer to [OBFrameMetadataType](#)

#### Returns

bool The result.

Definition at line [233](#) of file [Frame.hpp](#).

#### ◆ [getMetadataValue\(\)](#)

```
int64_t ob::Frame::getMetadataValue( OBFrameMetadataType type ) const
```

inline

Get the metadata value.

#### Parameters

**type** The metadata type. refer to [OBFrameMetadataType](#)

#### Returns

int64\_t The metadata value.

Definition at line [247](#) of file [Frame.hpp](#).

#### ◆ [getStreamProfile\(\)](#)

```
std::shared_ptr<StreamProfile> ob::Frame::getStreamProfile( ) const inline
```

get **StreamProfile** of the frame

#### Returns

std::shared\_ptr<StreamProfile> The **StreamProfile** of the frame, may return nullptr if the frame is not captured from a stream.

Definition at line **260** of file [Frame.hpp](#).

#### ◆ **getSensor()**

```
std::shared_ptr<Sensor> ob::Frame::getSensor( ) const inline
```

get owner sensor of the frame

#### Returns

std::shared\_ptr<Sensor> The owner sensor of the frame, return nullptr if the frame is not owned by any sensor or the sensor is destroyed

Definition at line **272** of file [Frame.hpp](#).

#### ◆ **getDevice()**

```
std::shared_ptr<Device> ob::Frame::getDevice( ) const inline
```

get owner device of the frame

#### Returns

std::shared\_ptr<Device> The owner device of the frame, return nullptr if the frame is not owned by any device or the device is destroyed

Definition at line **285** of file [Frame.hpp](#).

#### ◆ **is()**

```
template<typename T>
bool ob::Frame::is ( ) const
```

Check if the runtime type of the frame object is compatible with a given type.

### Template Parameters

**T**The given type.

### Returns

bool The result.

Definition at line [1051](#) of file [Frame.hpp](#).

Referenced by [as\(\)](#), [as\(\)](#), and [is\(\)](#).

## ◆ [as\(\)](#) [1/2]

```
template<typename T>
std::shared_ptr<T> ob::Frame::as ( )
```

inline

Convert the frame object to a target type.

### Template Parameters

**T**The target type.

### Returns

`std::shared_ptr<T>` The result. If it cannot be converted, an exception will be thrown.

Definition at line [307](#) of file [Frame.hpp](#).

## ◆ [as\(\)](#) [2/2]

```
template<typename T>
std::shared_ptr<const T> ob::Frame::as( ) const inline
```

Convert the frame object to a target type.

### Template Parameters

**T**The target type.

### Returns

`std::shared_ptr<T>` The result. If it cannot be converted, an exception will be thrown.

Definition at line [325](#) of file [Frame.hpp](#).

### ◆ `type()`

```
OBFrameType ob::Frame::type( ) const inline
```

Definition at line [339](#) of file [Frame.hpp](#).

Referenced by [getMetadataValue\(\)](#), [getType\(\)](#), and [hasMetadata\(\)](#).

### ◆ `format()`

```
virtual OBFormat ob::Frame::format( ) const inline virtual
```

Definition at line [343](#) of file [Frame.hpp](#).

Referenced by [getFormat\(\)](#).

### ◆ `index()`

```
virtual uint64_t ob::Frame::index( ) const inline virtual
```

Definition at line [347](#) of file [Frame.hpp](#).

Referenced by [ob::FrameSet::getFrame\(\)](#), [ob::FrameSet::getFrameByIndex\(\)](#), and [getIndex\(\)](#).

### ◆ `data()`

```
virtual void * ob::Frame::data( ) const inline virtual
```

Definition at line [351](#) of file [Frame.hpp](#).

Referenced by [data\(\)](#), and [getData\(\)](#).

◆ [dataSize\(\)](#)

```
virtual uint32_t ob::Frame::dataSize( ) const inline virtual
```

Definition at line [356](#) of file [Frame.hpp](#).

Referenced by [getDataSize\(\)](#).

◆ [timeStamp\(\)](#)

```
uint64_t ob::Frame::timeStamp( ) const inline
```

Definition at line [360](#) of file [Frame.hpp](#).

◆ [timeStampUs\(\)](#)

```
uint64_t ob::Frame::timeStampUs( ) const inline
```

Definition at line [364](#) of file [Frame.hpp](#).

Referenced by [getTimeStampUs\(\)](#).

◆ [systemTimeStamp\(\)](#)

```
uint64_t ob::Frame::systemTimeStamp( ) const inline
```

Definition at line [368](#) of file [Frame.hpp](#).

◆ [systemTimeStampUs\(\)](#)

```
uint64_t ob::Frame::systemTimeStampUs ( ) const
```

inline

Definition at line [372](#) of file [Frame.hpp](#).

Referenced by [getSystemTimeStampUs\(\)](#).

◆ [globalTimeStampUs\(\)](#)

```
uint64_t ob::Frame::globalTimeStampUs ( ) const
```

inline

Definition at line [376](#) of file [Frame.hpp](#).

Referenced by [getGlobalTimeStampUs\(\)](#).

◆ [metadata\(\)](#)

```
uint8_t * ob::Frame::metadata ( ) const
```

inline

Definition at line [380](#) of file [Frame.hpp](#).

Referenced by [getMetadata\(\)](#).

◆ [metadataSize\(\)](#)

```
uint32_t ob::Frame::metadataSize ( ) const
```

inline

Definition at line [384](#) of file [Frame.hpp](#).

Referenced by [getMetadataSize\(\)](#).

## Member Data Documentation

◆ [impl\\_](#)

```
const ob_frame* ob::Frame::impl_ = nullptr
```

protected

The pointer to the internal (c api level) frame object.

Definition at line **50** of file [Frame.hpp](#).

Referenced by [as\(\)](#), [as\(\)](#), [Frame\(\)](#), [ob::PointsFrame::getCoordinateValueScale\(\)](#), [ob::FrameSet::getCount\(\)](#), [getData\(\)](#), [getDataSize\(\)](#), [getDevice\(\)](#), [getFormat\(\)](#), [ob::FrameSet::getFrame\(\)](#), [ob::FrameSet::getFrameByIndex\(\)](#), [getGlobalTimeStampUs\(\)](#), [ob::PointsFrame::getHeight\(\)](#), [ob::VideoFrame::getHeight\(\)](#), [getImpl\(\)](#), [getIndex\(\)](#), [getMetadata\(\)](#), [getMetadataSize\(\)](#), [getMetadataValue\(\)](#), [ob::VideoFrame::getPixelAvailableBitSize\(\)](#), [ob::VideoFrame::getPixelType\(\)](#), [getSensor\(\)](#), [getStreamProfile\(\)](#), [getSystemTimeStampUs\(\)](#), [ob::AccelFrame::getTemperature\(\)](#), [ob::GyroFrame::getTemperature\(\)](#), [getTimeStampUs\(\)](#), [getType\(\)](#), [ob::AccelFrame::getValue\(\)](#), [ob::GyroFrame::getValue\(\)](#), [ob::DepthFrame::getValueScale\(\)](#), [ob::PointsFrame::getWidth\(\)](#), [ob::VideoFrame::getWidth\(\)](#), [hasMetadata\(\)](#), [ob::FrameSet::pushFrame\(\)](#), and [~Frame\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

## ob::FrameFactory Member List

This is the complete list of members for **ob::FrameFactory**, including all inherited members.

**BufferDestroyCallback** typedef

**createFrame**(OBFrameType frameType, OBFormat format, uint32\_t dataSize)

**createFrameFromBuffer**(OBFrameType frameType, OBFormat format, uint8\_t \*buffer, BufferDestroyCallbac

**createFrameFromOtherFrame**(std::shared\_ptr< const Frame > otherFrame, bool shouldCopyData=true)

**createFrameFromStreamProfile**(std::shared\_ptr< const StreamProfile > profile)

**createVideoFrame**(OBFrameType frameType, OBFormat format, uint32\_t width, uint32\_t height, uint32\_t stri

**createVideoFrameFromBuffer**(OBFrameType frameType, OBFormat format, uint32\_t width, uint32\_t height,

Generated on for OrbtecSDK by  1.14.0

# ob::FrameFactory Class Reference

**FrameFactory** class, which provides some static functions to create frame objects. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::FrameFactory:

## Public Types

```
typedef std::function< void(uint8_t *)> BufferDestroyCallback
```

The callback function to destroy the buffer when the frame is destroyed.

## Static Public Member Functions

```
static std::shared_ptr<Frame> createFrame (OBFrameType frameType, OBFormat format,  
uint32_t dataSize)
```

Create a **Frame** object of a specific type with a given format and data size.

```
static std::shared_ptr<VideoFrame> createVideoFrame (OBFrameType frameType, OBFormat  
format, uint32_t width, uint32_t height, uint32_t stride=0)
```

Create a **VideoFrame** object of a specific type with a given format, width, height, and stride.

```
static std::shared_ptr<Frame> createFrameFromOtherFrame (std::shared_ptr<const Frame>  
otherFrame, bool shouldCopyData=true)
```

Create (clone) a frame object based on the specified other frame object.

```
static std::shared_ptr<Frame> createFrameFromStreamProfile (std::shared_ptr<const  
StreamProfile> profile)
```

Create a **Frame** From (according to)Stream Profile object.

```
static std::shared_ptr<Frame> createFrameFromBuffer (OBFrameType frameType,  
OBFormat format, uint8_t *buffer, BufferDestroyCallback  
destroyCallback, uint32_t bufferSize)
```

Create a frame object based on an externally created buffer.

```
static std::shared_ptr<VideoFrame> createVideoFrameFromBuffer (OBFrameType frameType,  
OBFormat format, uint32_t width, uint32_t height, uint8_t  
*buffer, BufferDestroyCallback destroyCallback, uint32_t  
bufferSize, uint32_t stride=0)
```

Create a video frame object based on an externally created buffer.

## Detailed Description

**FrameFactory** class, which provides some static functions to create frame objects.

Definition at line 871 of file [Frame.hpp](#).

## Member Typedef Documentation

### ◆ BufferDestroyCallback

```
typedef std::function<void(uint8_t *)> ob::FrameFactory::BufferDestroyCallback
```

The callback function to destroy the buffer when the frame is destroyed.

Definition at line 946 of file [Frame.hpp](#).

## Member Function Documentation

### ◆ createFrame()

```
std::shared_ptr<Frame> ob::FrameFactory::createFrame ( OBFrameType frameType,  
                                              OBFormat      format,  
                                              uint32_t       dataSize )  
                                                 inline static
```

Create a **Frame** object of a specific type with a given format and data size.

#### Parameters

**frameType** The type of the frame.

**format** The format of the frame.

**dataSize** The size of the data in bytes.

#### Returns

`std::shared_ptr<Frame>` The created frame object.

Definition at line 881 of file [Frame.hpp](#).

### ◆ createVideoFrame()

```
std::shared_ptr<VideoFrame> ob::FrameFactory::createVideoFrame ( OBFrameType frameType,  
                                              OBFormat      format,  
                                              uint32_t       width,  
                                              uint32_t       height,  
                                              uint32_t       stride = 0 )           inline static
```

Create a **VideoFrame** object of a specific type with a given format, width, height, and stride.

### Note

If stride is not specified, it will be calculated based on the width and format.

### Parameters

**frameType** The type of the frame.  
**format** The format of the frame.  
**width** The width of the frame.  
**height** The height of the frame.  
**stride** The stride of the frame.

### Returns

std::shared\_ptr<VideoFrame> The created video frame object.

Definition at line **901** of file **Frame.hpp**.

◆ [createFrameFromOtherFrame\(\)](#)

```
std::shared_ptr<Frame>
ob::FrameFactory::createFrameFromOtherFrame ( std::shared_ptr<const Frame> otherFrame,
                                              bool                                shouldCopyData = true )  inline  stat
```

Create (clone) a frame object based on the specified other frame object.

The new frame object will have the same properties as the other frame object, but the data buffer is newly allocated.

### Parameters

**shouldCopyData** If true, the data of the source frame object will be copied to the new frame object. If false, the new frame object will have a data buffer with random data. The default value is true.

### Returns

std::shared\_ptr<Frame> The new frame object.

Definition at line [919](#) of file [Frame.hpp](#).

## ◆ [createFrameFromStreamProfile\(\)](#)

```
std::shared_ptr<Frame>
ob::FrameFactory::createFrameFromStreamProfile ( std::shared_ptr<const StreamProfile> profile )  inline  static
```

Create a **Frame** From (according to)Stream Profile object.

### Parameters

**profile** The stream profile object to create the frame from.

### Returns

std::shared\_ptr<Frame> The created frame object.

Definition at line [935](#) of file [Frame.hpp](#).

## ◆ [createFrameFromBuffer\(\)](#)

```
std::shared_ptr<Frame>
ob::FrameFactory::createFrameFromBuffer
( OBFrameType frameType,
  OBFormat format,
  uint8_t * buffer,
  BufferDestroyCallback destroyCallback,
  uint32_t bufferSize ) inline static
```

Create a frame object based on an externally created buffer.

### Attention

The buffer is owned by the caller, and will not be destroyed by the frame object. The user should ensure that the buffer is valid and not modified.

### Parameters

[in] <b>frameType</b>	<b>Frame</b> object type.
[in] <b>format</b>	<b>Frame</b> object format.
[in] <b>buffer</b>	<b>Frame</b> object buffer.
[in] <b>destroyCallback</b>	Destroy callback, will be called when the frame object is destroyed.
[in] <b>bufferSize</b>	<b>Frame</b> object buffer size.

### Returns

std::shared\_ptr<Frame> The created frame object.

Definition at line [962](#) of file [Frame.hpp](#).

- ◆ [createVideoFrameFromBuffer\(\)](#)

```

std::shared_ptr<VideoFrame>
ob::FrameFactory::createVideoFrameFromBuffer
    ( OBFrameType frameType,
      OBFormat format,
      uint32_t width,
      uint32_t height,
      uint8_t * buffer,
      BufferDestroyCallback destroyCallback,
      uint32_t bufferSize,
      uint32_t stride = 0 )           inline static

```

Create a video frame object based on an externally created buffer.

### Attention

The buffer is owned by the user and will not be destroyed by the frame object. The user should ensure that the buffer is valid and not modified.

The frame object is created with a reference count of 1, and the reference count should be decreased by calling [ob\\_delete\\_frame\(\)](#) when it is no longer needed.

### Parameters

[in] <b>frameType</b>	<a href="#">Frame</a> object type.
[in] <b>format</b>	<a href="#">Frame</a> object format.
[in] <b>width</b>	<a href="#">Frame</a> object width.
[in] <b>height</b>	<a href="#">Frame</a> object height.
[in] <b>buffer</b>	<a href="#">Frame</a> object buffer.
[in] <b>bufferSize</b>	<a href="#">Frame</a> object buffer size.
[in] <b>destroyCallback</b>	Destroy callback, will be called when the frame object is destroyed.
[in] <b>stride</b>	Row span in bytes. If 0, the stride is calculated based on the width and format.

### Returns

`std::shared_ptr<VideoFrame>` The created video frame object.

Definition at line [991](#) of file [Frame.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

## ob::FrameHelper Member List

This is the complete list of members for **ob::FrameHelper**, including all inherited members.

**BufferDestroyCallback** typedef

**createFrame**(OBFrameType frameType, OBFormat format, uint32\_t dataSize)

**createFrameFromBuffer**(OBFrameType frameType, OBFormat format, uint8\_t \*buffer, BufferDestroyCallback)

**createFrameFromOtherFrame**(std::shared\_ptr< const Frame > otherFrame, bool shouldCopyData=true)

**createFrameFromStreamProfile**(std::shared\_ptr< const StreamProfile > profile)

**createVideoFrame**(OBFrameType frameType, OBFormat format, uint32\_t width, uint32\_t height, uint32\_t strikeWidth, uint32\_t strikeHeight)

**createVideoFrameFromBuffer**(OBFrameType frameType, OBFormat format, uint32\_t width, uint32\_t height, uint8\_t \*buffer, BufferDestroyCallback)

**setFrameDeviceTimestamp**(std::shared\_ptr< Frame > frame, uint64\_t deviceTimestamp)

**setFrameDeviceTimestampUs**(std::shared\_ptr< Frame > frame, uint64\_t deviceTimestampUs)

**setFrameSystemTimestamp**(std::shared\_ptr< Frame > frame, uint64\_t systemTimestamp)

# ob::FrameHelper Class Reference

FrameHepler class, which provides some static functions to set timestamp for frame objects FrameHepler inherited from the **FrameFactory** and the timestamp interface implement here both for compatibility purposes. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::FrameHelper:

## Static Public Member Functions

```
static void setFrameDeviceTimestampUs (std::shared_ptr<Frame> frame, uint64_t deviceTimestampUs)
```

Set the device timestamp of the frame.

```
static void setFrameSystemTimestamp (std::shared_ptr<Frame> frame, uint64_t systemTimestamp)
```

```
static void setFrameDeviceTimestamp (std::shared_ptr<Frame> frame, uint64_t deviceTimestamp)
```

Static Public Member Functions inherited from **ob::FrameFactory**

## Additional Inherited Members

Public Types inherited from **ob::FrameFactory**

## Detailed Description

FrameHepler class, which provides some static functions to set timestamp for frame objects FrameHepler inherited from the **FrameFactory** and the timestamp interface implement here both for compatibility purposes.

Definition at line **1020** of file **Frame.hpp**.

## Member Function Documentation

◆ **setFrameDeviceTimestampUs()**

```
void ob::FrameHelper::setFrameDeviceTimestampUs ( std::shared_ptr<Frame> frame,
                                                uint64_t deviceTimestampUs ) inline static
```

Set the device timestamp of the frame.

#### Parameters

**frame** The frame object.

**deviceTimestampUs** The device timestamp to set in microseconds.

Definition at line [1028](#) of file [Frame.hpp](#).

#### ◆ [setFrameSystemTimestamp\(\)](#)

```
void ob::FrameHelper::setFrameSystemTimestamp ( std::shared_ptr<Frame> frame,
                                                uint64_t systemTimestamp ) inline static
```

Definition at line [1037](#) of file [Frame.hpp](#).

#### ◆ [setFrameDeviceTimestamp\(\)](#)

```
void ob::FrameHelper::setFrameDeviceTimestamp ( std::shared_ptr<Frame> frame,
                                                uint64_t deviceTimestamp ) inline static
```

Definition at line [1043](#) of file [Frame.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

## ob::FrameSet Member List

This is the complete list of members for **ob::FrameSet**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>colorFrame() const</b>	<b>ob::FrameSet</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>depthFrame() const</b>	<b>ob::FrameSet</b>	inline
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>frameCount() const</b>	<b>ob::FrameSet</b>	inline
<b>FrameSet(const ob_frame *impl)</b>	<b>ob::FrameSet</b>	inline explicit
<b>getCount() const</b>	<b>ob::FrameSet</b>	inline
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getFrame(OBFrameType frameType) const</b>	<b>ob::FrameSet</b>	inline
<b>getFrame(int index) const</b>	<b>ob::FrameSet</b>	inline
<b>getFrameByIndex(uint32_t index) const</b>	<b>ob::FrameSet</b>	inline
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected

<b>index()</b> const	<b>ob::Frame</b>	inline virtual
<b>irFrame()</b> const	<b>ob::FrameSet</b>	inline
<b>is()</b> const	<b>ob::Frame</b>	
<b>metadata()</b> const	<b>ob::Frame</b>	inline
<b>metadataSize()</b> const	<b>ob::Frame</b>	inline
<b>pointsFrame()</b> const	<b>ob::FrameSet</b>	inline
<b>pushFrame</b> (std::shared_ptr<const Frame> frame) const	<b>ob::FrameSet</b>	inline
<b>systemTimeStamp()</b> const	<b>ob::Frame</b>	inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~FrameSet()</b> noexcept override=default	<b>ob::FrameSet</b>	

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::FrameSet Class Reference

Define the **FrameSet** class, which inherits from the **Frame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::FrameSet:

## Public Member Functions

```
FrameSet (const ob_frame *impl)  
~FrameSet () noexcept override=default  
uint32_t getCount () const  
    Get the number of frames in the FrameSet.  
std::shared_ptr<Frame> getFrame (OBFrameType frameType) const  
    Get a frame of a specific type from the FrameSet.  
std::shared_ptr<Frame> getFrameByIndex (uint32_t index) const  
    Get a frame at a specific index from the FrameSet.  
void pushFrame (std::shared_ptr<const Frame> frame) const  
    Push a frame to the FrameSet.  
    uint32_t frameCount () const  
std::shared_ptr<DepthFrame> depthFrame () const  
std::shared_ptr<ColorFrame> colorFrame () const  
std::shared_ptr<IRFrame> irFrame () const  
std::shared_ptr<PointsFrame> pointsFrame () const  
std::shared_ptr<Frame> getFrame (int index) const
```

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **FrameSet** class, which inherits from the **Frame** class.

A **FrameSet** is a container for multiple frames of different types.

Definition at line **749** of file **Frame.hpp**.

## Constructor & Destructor Documentation

### ◆ FrameSet()

ob::FrameSet::FrameSet ( const **ob\_frame** \* impl )

inline explicit

Definition at line [752](#) of file **Frame.hpp**.

### ◆ ~FrameSet()

ob::FrameSet::~FrameSet ( )

override default noexcept

## Member Function Documentation

### ◆ getCount()

uint32\_t ob::FrameSet::getCount ( ) const

inline

Get the number of frames in the **FrameSet**.

#### Returns

uint32\_t The number of frames

Definition at line [761](#) of file **Frame.hpp**.

Referenced by **frameCount()**, and **getCount()**.

### ◆ getFrame0 [1/2]

```
std::shared_ptr<Frame> ob::FrameSet::getFrame ( OBFrameType frameType ) const inline
```

Get a frame of a specific type from the **FrameSet**.

#### Parameters

**frameType** The type of sensor

#### Returns

std::shared\_ptr<Frame> The corresponding type of frame

Definition at line [774](#) of file **Frame.hpp**.

Referenced by **colorFrame()**, **depthFrame()**, **irFrame()**, and **pointsFrame()**.

### ◆ **getFrameByIndex()**

```
std::shared_ptr<Frame> ob::FrameSet::getFrameByIndex ( uint32_t index ) const inline
```

Get a frame at a specific index from the **FrameSet**.

#### Parameters

**index** The index of the frame

#### Returns

std::shared\_ptr<Frame> The frame at the specified index

Definition at line [790](#) of file **Frame.hpp**.

Referenced by **getFrame()**.

### ◆ **pushFrame()**

```
void ob::FrameSet::pushFrame ( std::shared_ptr<const Frame> frame ) const inline
```

Push a frame to the **FrameSet**.

#### Attention

If the **FrameSet** contains the same type of frame, the new frame will replace the old one.

#### Parameters

**frame** The frame to be pushed

Definition at line [807](#) of file **Frame.hpp**.

◆ **frameCount()**

uint32\_t ob::FrameSet::frameCount ( ) const

inline

Definition at line [821](#) of file [Frame.hpp](#).

◆ **depthFrame()**

std::shared\_ptr<[DepthFrame](#)> ob::FrameSet::depthFrame ( ) const

inline

Definition at line [825](#) of file [Frame.hpp](#).

Referenced by [depthFrame\(\)](#).

◆ **colorFrame()**

std::shared\_ptr<[ColorFrame](#)> ob::FrameSet::colorFrame ( ) const

inline

Definition at line [834](#) of file [Frame.hpp](#).

Referenced by [colorFrame\(\)](#).

◆ **irFrame()**

std::shared\_ptr<[IRFrame](#)> ob::FrameSet::irFrame ( ) const

inline

Definition at line [843](#) of file [Frame.hpp](#).

Referenced by [irFrame\(\)](#).

◆ **pointsFrame()**

std::shared\_ptr<[PointsFrame](#)> ob::FrameSet::pointsFrame ( ) const

inline

Definition at line [854](#) of file [Frame.hpp](#).

Referenced by [pointsFrame\(\)](#).

◆ [getFrame\(\)](#) [2/2]

std::shared\_ptr<[Frame](#)> ob::FrameSet::getFrame ( int index ) const

inline

Definition at line [863](#) of file [Frame.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::GyroFrame Member List

This is the complete list of members for **ob::GyroFrame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTemperature() const</b>	<b>ob::GyroFrame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getValue() const</b>	<b>ob::GyroFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>GyroFrame(const ob_frame *impl)</b>	<b>ob::GyroFrame</b>	inline explicit
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	
<b>metadata() const</b>	<b>ob::Frame</b>	inline
<b>metadataSize() const</b>	<b>ob::Frame</b>	inline
<b>systemTimeStamp() const</b>	<b>ob::Frame</b>	inline

<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>temperature()</b>	<b>ob::GyroFrame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>value()</b>	<b>ob::GyroFrame</b>	inline
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~GyroFrame()</b> noexcept override=default	<b>ob::GyroFrame</b>	

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::GyroFrame Class Reference

Define the **GyroFrame** class, which inherits from the **Frame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::GyroFrame:

## Public Member Functions

```
GyroFrame (const ob_frame *impl)  
~GyroFrame () noexcept override=default
```

```
OBGyroValue getValue () const
```

Get the gyro frame data.

```
float getTemperature () const
```

Get the temperature when the frame was sampled.

```
OBGyroValue value ()
```

```
float temperature ()
```

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **GyroFrame** class, which inherits from the **Frame** class.

Definition at line **701** of file **Frame.hpp**.

## Constructor & Destructor Documentation

### ◆ GyroFrame()

```
ob::GyroFrame::GyroFrame ( const ob_frame * impl )
```

inline explicit

Definition at line **704** of file **Frame.hpp**.

◆ ~GyroFrame()

ob::GyroFrame::~GyroFrame ( )

override default noexcept

## Member Function Documentation

◆ getValue()

**OBGyroValue** ob::GyroFrame::getValue ( ) const

inline

Get the gyro frame data.

**Returns**

**OBAccelValue** The gyro frame data

Definition at line [713](#) of file **Frame.hpp**.

Referenced by [getValue\(\)](#), and [value\(\)](#).

◆ getTemperature()

float ob::GyroFrame::getTemperature ( ) const

inline

Get the temperature when the frame was sampled.

**Returns**

float The temperature value in celsius

Definition at line [726](#) of file **Frame.hpp**.

Referenced by [temperature\(\)](#).

◆ value()

**OBGyroValue** ob::GyroFrame::value ( )

inline

Definition at line [736](#) of file **Frame.hpp**.

Referenced by [getValue\(\)](#).

◆ **temperature()**

float ob::GyroFrame::temperature ( )

inline

Definition at line **740** of file [Frame.hpp](#).

Referenced by [getTemperature\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::GyroStreamProfile Member List

This is the complete list of members for **ob::GyroStreamProfile**, including all inherited members.

<b>as()</b>	<b>ob::StreamProfile</b>
<b>as() const</b>	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo(std::shared_ptr&lt; StreamProfile &gt; target, const OBExtrinsic &amp;extrinsic)</b>	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo(const OBStreamType &amp;targetStreamType, const OBExtrinsic &amp;extrinsic)</b>	<b>ob::StreamProfile</b>
<b>format() const</b>	<b>ob::StreamProfile</b>
<b>fullScaleRange() const</b>	<b>ob::GyroStreamProfile</b>
<b>getExtrinsicTo(std::shared_ptr&lt; StreamProfile &gt; target) const</b>	<b>ob::StreamProfile</b>
<b>getFormat() const</b>	<b>ob::StreamProfile</b>
<b>getFullScaleRange() const</b>	<b>ob::GyroStreamProfile</b>
<b>getImpl() const</b>	<b>ob::StreamProfile</b>
<b>getIntrinsic() const</b>	<b>ob::GyroStreamProfile</b>
<b>getSampleRate() const</b>	<b>ob::GyroStreamProfile</b>
<b>getType() const</b>	<b>ob::StreamProfile</b>
<b>GyroStreamProfile(const ob_stream_profile_t *impl)</b>	<b>ob::GyroStreamProfile</b>
<b>impl_</b>	<b>ob::StreamProfile</b>
<b>is() const</b>	<b>ob::StreamProfile</b>
<b>operator=(StreamProfile &amp;streamProfile)=delete</b>	<b>ob::StreamProfile</b>
<b>operator=(StreamProfile &amp;&amp;streamProfile) noexcept</b>	<b>ob::StreamProfile</b>
<b>sampleRate() const</b>	<b>ob::GyroStreamProfile</b>
<b>StreamProfile(StreamProfile &amp;streamProfile)=delete</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(StreamProfile &amp;&amp;streamProfile) noexcept</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(const ob_stream_profile_t *impl)</b>	<b>ob::StreamProfile</b>
<b>type() const</b>	<b>ob::StreamProfile</b>
<b>~GyroStreamProfile() noexcept override=default</b>	<b>ob::GyroStreamProfile</b>
<b>~StreamProfile() noexcept</b>	<b>ob::StreamProfile</b>

# ob::GyroStreamProfile Class Reference

Class representing a gyroscope stream profile. [More...](#)

```
#include <StreamProfile.hpp>
```

Inheritance diagram for ob::GyroStreamProfile:

## Public Member Functions

**GyroStreamProfile** (const ob\_stream\_profile\_t \*impl)

**~GyroStreamProfile** () noexcept override=default

**OBGyroFullScaleRange** **getFullScaleRange** () const

Return the full scale range.

**OBGyroSampleRate** **getSampleRate** () const

Return the sampling frequency.

**OBGyroIntrinsic** **getIntrinsic** () const

get the intrinsic parameters of the stream.

**OBGyroFullScaleRange** **fullScaleRange** () const

**OBGyroSampleRate** **sampleRate** () const

Public Member Functions inherited from **ob::StreamProfile**

## Additional Inherited Members

Protected Member Functions inherited from **ob::StreamProfile**

Protected Attributes inherited from **ob::StreamProfile**

## Detailed Description

Class representing a gyroscope stream profile.

Definition at line **328** of file **StreamProfile.hpp**.

## Constructor & Destructor Documentation

- ◆ **GyroStreamProfile()**

```
ob::GyroStreamProfile::GyroStreamProfile ( const ob_stream_profile_t * impl )
```

inline explicit

Definition at line [330](#) of file [StreamProfile.hpp](#).

◆ ~GyroStreamProfile()

```
ob::GyroStreamProfile::~GyroStreamProfile ( )
```

override default noexcept

## Member Function Documentation

◆ getFullScaleRange()

**OBGyroFullScaleRange** ob::GyroStreamProfile::getFullScaleRange ( ) const

inline

Return the full scale range.

**Returns**

**OBAccelFullScaleRange** Return the scale range value.

Definition at line [339](#) of file [StreamProfile.hpp](#).

Referenced by [fullScaleRange\(\)](#), and [getFullScaleRange\(\)](#).

◆ getSampleRate()

**OBGyroSampleRate** ob::GyroStreamProfile::getSampleRate ( ) const

inline

Return the sampling frequency.

**Returns**

**OBAccelFullScaleRange** Return the sampling frequency.

Definition at line [351](#) of file [StreamProfile.hpp](#).

Referenced by [sampleRate\(\)](#).

◆ getIntrinsic()

**OBGyroIntrinsic** ob::GyroStreamProfile::getIntrinsic ( ) const

inline

get the intrinsic parameters of the stream.

### Returns

**OBGyroIntrinsic** Return the intrinsic parameters.

Definition at line **363** of file **StreamProfile.hpp**.

### ◆ fullScaleRange()

**OBGyroFullScaleRange** ob::GyroStreamProfile::fullScaleRange ( ) const

inline

Definition at line **372** of file **StreamProfile.hpp**.

Referenced by [getFullScaleRange\(\)](#).

### ◆ sampleRate()

**OBGyroSampleRate** ob::GyroStreamProfile::sampleRate ( ) const

inline

Definition at line **376** of file **StreamProfile.hpp**.

Referenced by [getSampleRate\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**StreamProfile.hpp**

# ob::HdrMerge Member List

This is the complete list of members for **ob::HdrMerge**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>HdrMerge()</b>	<b>ob::HdrMerge</b>	inline
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virtual
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt;const Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt;Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~HdrMerge() noexcept override=default</b>	<b>ob::HdrMerge</b>	virtual

# ob::HdrMerge Class Reference

**HdrMerge** processing block, the processing merges between depth frames with different sub-preset sequence ids. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::HdrMerge:

## Public Member Functions

**HdrMerge ()**

virtual **~HdrMerge ()** noexcept override=default

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

**HdrMerge** processing block, the processing merges between depth frames with different sub-preset sequence ids.

Definition at line [486](#) of file [Filter.hpp](#).

## Constructor & Destructor Documentation

◆ **HdrMerge()**

ob::HdrMerge::HdrMerge ( )

inline

Definition at line [488](#) of file [Filter.hpp](#).

◆ **~HdrMerge()**

virtual ob::HdrMerge::~HdrMerge ( )

override virtual default noexcept

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::HoleFillingFilter Member List

This is the complete list of members for **ob::HoleFillingFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getFilterMode()</b>	<b>ob::HoleFillingFilter</b>	inline
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>HoleFillingFilter(const std::string &amp;activationKey="")</b>	<b>ob::HoleFillingFilter</b>	inline
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virtual
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt;const Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt;Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setFilterMode(OBHoleFillingMode mode)</b>	<b>ob::HoleFillingFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~HoleFillingFilter() noexcept override=default</b>	<b>ob::HoleFillingFilter</b>	

# ob::HoleFillingFilter Class Reference

Hole filling filter, the processing performed depends on the selected hole filling mode. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::HoleFillingFilter:

## Public Member Functions

```
HoleFillingFilter (const std::string &activationKey="")
```

```
-HoleFillingFilter () noexcept override=default
```

```
void setFilterMode (OBHoleFillingMode mode)
```

Set the **HoleFillingFilter** mode.

```
OBHoleFillingMode getFilterMode ()
```

Get the **HoleFillingFilter** mode.

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

Hole filling filter, the processing performed depends on the selected hole filling mode.

Definition at line [785](#) of file **Filter.hpp**.

## Constructor & Destructor Documentation

### ◆ **HoleFillingFilter()**

```
ob::HoleFillingFilter::HoleFillingFilter ( const std::string & activationKey = "" )
```

inline

Definition at line [787](#) of file **Filter.hpp**.

### ◆ **~HoleFillingFilter()**

```
ob::HoleFillingFilter::~HoleFillingFilter( ) override default noexcept
```

## Member Function Documentation

### ◆ setFilterMode()

```
void ob::HoleFillingFilter::setFilterMode ( OBHoleFillingMode mode ) inline
```

Set the **HoleFillingFilter** mode.

#### Parameters

**mode** **OBHoleFillingMode**, OB\_HOLE\_FILL\_TOP,OB\_HOLE\_FILL\_NEAREST or  
OB\_HOLE\_FILL\_FADEST.

Definition at line **801** of file **Filter.hpp**.

Referenced by **setFilterMode()**.

### ◆ getFilterMode()

```
OBHoleFillingMode ob::HoleFillingFilter::getFilterMode ( ) inline
```

Get the **HoleFillingFilter** mode.

#### Returns

**OBHoleFillingMode**

Definition at line **810** of file **Filter.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

## ob::IRFrame Member List

This is the complete list of members for **ob::IRFrame**, including all inherited members.

<a href="#">as()</a>	<b>ob::Frame</b>	inline
<a href="#">as() const</a>	<b>ob::Frame</b>	inline
<a href="#">data() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">dataSize() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">format() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">Frame(const ob_frame *impl)</a>	<b>ob::Frame</b>	inline explicit
<a href="#">getData() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">getDataSize() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">getDevice() const</a>	<b>ob::Frame</b>	inline
<a href="#">getFormat() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">getGlobalTimeStampUs() const</a>	<b>ob::Frame</b>	inline
<a href="#">getHeight() const</a>	<b>ob::VideoFrame</b>	inline
<a href="#">getImpl() const</a>	<b>ob::Frame</b>	inline
<a href="#">getIndex() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">getMetadata() const</a>	<b>ob::Frame</b>	inline
<a href="#">getMetadataSize() const</a>	<b>ob::Frame</b>	inline
<a href="#">getMetadataValue(OBFrameMetadataType type) const</a>	<b>ob::Frame</b>	inline
<a href="#">getPixelAvailableBitSize() const</a>	<b>ob::VideoFrame</b>	inline
<a href="#">getPixelType() const</a>	<b>ob::VideoFrame</b>	inline
<a href="#">getSensor() const</a>	<b>ob::Frame</b>	inline
<a href="#">getStreamProfile() const</a>	<b>ob::Frame</b>	inline
<a href="#">getSystemTimeStampUs() const</a>	<b>ob::Frame</b>	inline
<a href="#">getTimeStampUs() const</a>	<b>ob::Frame</b>	inline
<a href="#">getType() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">getWidth() const</a>	<b>ob::VideoFrame</b>	inline
<a href="#">globalTimeStampUs() const</a>	<b>ob::Frame</b>	inline
<a href="#">hasMetadata(OBFrameMetadataType type) const</a>	<b>ob::Frame</b>	inline
<a href="#">height() const</a>	<b>ob::VideoFrame</b>	inline
<a href="#">impl_</a>	<b>ob::Frame</b>	protected
<a href="#">index() const</a>	<b>ob::Frame</b>	inline virtual
<a href="#">IRFrame(const ob_frame *impl)</a>	<b>ob::IRFrame</b>	inline explicit
<a href="#">is() const</a>	<b>ob::Frame</b>	

<b>metadata()</b> const	<b>ob::Frame</b>	inline
<b>metadataSize()</b> const	<b>ob::Frame</b>	inline
<b>pixelAvailableBitSize()</b> const	<b>ob::VideoFrame</b>	inline
<b>systemTimeStamp()</b> const	<b>ob::Frame</b>	inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>VideoFrame</b> (const ob_frame *impl)	<b>ob::VideoFrame</b>	inline explicit
<b>width()</b> const	<b>ob::VideoFrame</b>	inline
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~IRFrame()</b> noexcept override=default	<b>ob::IRFrame</b>	
<b>~VideoFrame()</b> noexcept override=default	<b>ob::VideoFrame</b>	

# ob::IRFrame Class Reference

Define the **IRFrame** class, which inherits from the **VideoFrame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::IRFrame:

## Public Member Functions

**IRFrame** (const **ob\_frame** \*impl)

Construct a new **IRFrame** object with a given pointer to the internal frame object.

**~IRFrame** () noexcept override=default

Public Member Functions inherited from **ob::VideoFrame**

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **IRFrame** class, which inherits from the **VideoFrame** class.

Definition at line **538** of file **Frame.hpp**.

## Constructor & Destructor Documentation

- ◆ **IRFrame()**

```
ob::IRFrame::IRFrame ( const ob_frame * impl )  
    inline explicit
```

Construct a new **IRFrame** object with a given pointer to the internal frame object.

### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

Please use the **FrameFactory** to create a **Frame** object.

### Parameters

**impl** The pointer to the internal frame object.

Definition at line **551** of file **Frame.hpp**.

### ◆ ~IRFrame()

```
ob::IRFrame::~IRFrame ( )  
    override default noexcept
```

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Frame.hpp**

# ob::NoiseRemovalFilter Member List

This is the complete list of members for **ob::NoiseRemovalFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getDispDiffRange()</b>	<b>ob::NoiseRemovalFilter</b>	inline
<b>getFilterParams()</b>	<b>ob::NoiseRemovalFilter</b>	inline
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getMaxSizeRange()</b>	<b>ob::NoiseRemovalFilter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>NoiseRemovalFilter(const std::string &amp;activationKey="")</b>	<b>ob::NoiseRemovalFilter</b>	inline
<b>process(std::shared_ptr&lt;const Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt;Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setFilterParams(OBNoiseRemovalFilterParams filterParams)</b>	<b>ob::NoiseRemovalFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~NoiseRemovalFilter() noexcept override=default</b>	<b>ob::NoiseRemovalFilter</b>	

# ob::NoiseRemovalFilter Class Reference

The noise removal filter,removing scattering depth pixels. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::NoiseRemovalFilter:

## Public Member Functions

**NoiseRemovalFilter** (const std::string &activationKey="")

**~NoiseRemovalFilter** () noexcept override=default

void **setFilterParams** (**OBNoiseRemovalFilterParams** filterParams)

Set the noise removal filter params.

**OBNoiseRemovalFilterParams getFilterParams ()**

Get the noise removal filter params.

**OBUInt16PropertyRange getDispDiffRange ()**

Get the noise removal filter disp diff range.

**OBUInt16PropertyRange getMaxSizeRange ()**

Get the noise removal filter max size range.

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

The noise removal filter,removing scattering depth pixels.

Definition at line **818** of file **Filter.hpp**.

## Constructor & Destructor Documentation

◆ **NoiseRemovalFilter()**

```
ob::NoiseRemovalFilter::NoiseRemovalFilter ( const std::string & activationKey = "" ) inline
```

Definition at line [820](#) of file [Filter.hpp](#).

◆ ~NoiseRemovalFilter()

```
ob::NoiseRemovalFilter::~NoiseRemovalFilter ( ) override default noexcept
```

## Member Function Documentation

◆ setFilterParams()

```
void ob::NoiseRemovalFilter::setFilterParams ( OBNoiseRemovalFilterParams filterParams ) inline
```

Set the noise removal filter params.

### Parameters

[in] **filterParams** [ob\\_noise\\_removal\\_filter\\_params](#).

Definition at line [834](#) of file [Filter.hpp](#).

Referenced by [setFilterParams\(\)](#).

◆ getFilterParams()

```
OBNoiseRemovalFilterParams ob::NoiseRemovalFilter::getFilterParams ( ) inline
```

Get the noise removal filter params.

### Returns

[OBNoiseRemovalFilterParams](#).

Definition at line [845](#) of file [Filter.hpp](#).

◆ getDispDiffRange()

**OB UInt16PropertyRange** ob::NoiseRemovalFilter::getDispDiffRange ( )

inline

Get the noise removal filter disp diff range.

**Returns**

**OB UInt16PropertyRange** The disp diff of property range.

Definition at line [857](#) of file [Filter.hpp](#).

◆ [getMaxSizeRange\(\)](#)

**OB UInt16PropertyRange** ob::NoiseRemovalFilter::getMaxSizeRange ( )

inline

Get the noise removal filter max size range.

**Returns**

**OB UInt16PropertyRange** The max size of property range.

Definition at line [873](#) of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::OBDepthWorkModeList Member List

This is the complete list of members for **ob::OBDepthWorkModeList**, including all inherited members.

<b>count()</b>	<b>ob::OBDepthWorkModeList</b>	inline
<b>getCount()</b>	<b>ob::OBDepthWorkModeList</b>	inline
<b>getOBDepthWorkMode(uint32_t index)</b>	<b>ob::OBDepthWorkModeList</b>	inline
<b>OBDepthWorkModeList(ob_depth_work_mode_list_t *impl)</b>	<b>ob::OBDepthWorkModeList</b>	inline explicit
<b>operator[](uint32_t index)</b>	<b>ob::OBDepthWorkModeList</b>	inline
<b>~OBDepthWorkModeList()</b>	<b>ob::OBDepthWorkModeList</b>	inline

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::OBDepthWorkModeList Class Reference

```
#include <Device.hpp>
```

## Public Member Functions

**OBDepthWorkModeList** (ob\_depth\_work\_mode\_list\_t \*impl)

**~OBDepthWorkModeList** ()

uint32\_t **getCount** ()

Get the number of **OBDepthWorkMode** objects in the list.

**OBDepthWorkMode** **getOBDepthWorkMode** (uint32\_t index)

Get the **OBDepthWorkMode** object at the specified index.

**OBDepthWorkMode** **operator[]** (uint32\_t index)

Get the **OBDepthWorkMode** object at the specified index.

uint32\_t **count** ()

## Detailed Description

Definition at line [1304](#) of file [Device.hpp](#).

## Constructor & Destructor Documentation

◆ **OBDepthWorkModeList()**

ob::OBDepthWorkModeList::OBDepthWorkModeList ( ob\_depth\_work\_mode\_list\_t \* impl )

inline explicit

Definition at line [1309](#) of file [Device.hpp](#).

◆ **~OBDepthWorkModeList()**

ob::OBDepthWorkModeList::~OBDepthWorkModeList ( )

inline

Definition at line [1310](#) of file [Device.hpp](#).

## Member Function Documentation

◆ **getCount()**

`uint32_t ob::OBDepthWorkModeList::getCount( )`

inline

Get the number of **OBDepthWorkMode** objects in the list.

**Returns**

`uint32_t` the number of **OBDepthWorkMode** objects in the list

Definition at line [1321](#) of file **Device.hpp**.

Referenced by **count()**.

◆ **getOBDepthWorkMode()**

**OBDepthWorkMode** `ob::OBDepthWorkModeList::getOBDepthWorkMode( uint32_t index )`

inline

Get the **OBDepthWorkMode** object at the specified index.

**Parameters**

`index` the index of the target **OBDepthWorkMode** object

**Returns**

**OBDepthWorkMode** the **OBDepthWorkMode** object at the specified index

Definition at line [1334](#) of file **Device.hpp**.

Referenced by **operator[]()**.

◆ **operator[]()**

**OBDepthWorkMode** `ob::OBDepthWorkModeList::operator[]( uint32_t index )`

inline

Get the **OBDepthWorkMode** object at the specified index.

**Parameters**

`index` the index of the target **OBDepthWorkMode** object

**Returns**

**OBDepthWorkMode** the **OBDepthWorkMode** object at the specified index

Definition at line [1347](#) of file **Device.hpp**.

◆ **count()**

uint32\_t ob::OBDepthWorkModeList::count ( )

inline

Definition at line [1353](#) of file **Device.hpp**.

Referenced by [getCount\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Device.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::OBFilterList Member List

This is the complete list of members for **ob::OBFilterList**, including all inherited members.

<b>count()</b> const	<b>ob::OBFilterList</b>	inline
<b>getCount()</b> const	<b>ob::OBFilterList</b>	inline
<b>getFilter(uint32_t index)</b>	<b>ob::OBFilterList</b>	inline
<b>OBFilterList(ob_filter_list_t *impl)</b>	<b>ob::OBFilterList</b>	inline explicit
<b>~OBFilterList()</b> noexcept	<b>ob::OBFilterList</b>	inline

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::OBFilterList Class Reference

```
#include <Filter.hpp>
```

## Public Member Functions

```
OBFilterList (ob_filter_list_t *impl)  
~OBFilterList () noexcept  
    uint32_t getCount () const  
        Get the number of filters.  
    std::shared_ptr<Filter > getFilter (uint32_t index)  
        Get the Filter object at the specified index.  
    uint32_t count () const
```

## Detailed Description

Definition at line [968](#) of file [Filter.hpp](#).

## Constructor & Destructor Documentation

### ◆ OBFilterList()

```
ob::OBFilterList::OBFilterList ( ob_filter_list_t * impl )  
    inline explicit
```

Definition at line [973](#) of file [Filter.hpp](#).

### ◆ ~OBFilterList()

```
ob::OBFilterList::~OBFilterList ( )  
    inline noexcept
```

Definition at line [975](#) of file [Filter.hpp](#).

## Member Function Documentation

### ◆ getCount()

uint32\_t ob::OBFilterList::getCount ( ) const

inline

Get the number of filters.

#### Returns

uint32\_t The number of filters

Definition at line **986** of file [Filter.hpp](#).

Referenced by [count\(\)](#).

### ◆ getFilter()

std::shared\_ptr<[Filter](#)> ob::OBFilterList::getFilter ( uint32\_t index )

inline

Get the [Filter](#) object at the specified index.

#### Parameters

**index** The filter index. The range is [0, count-1]. If the index exceeds the range, an exception will be thrown.

#### Returns

std::shared\_ptr<Filter> The filter object.

Definition at line **999** of file [Filter.hpp](#).

### ◆ count()

uint32\_t ob::OBFilterList::count ( ) const

inline

Definition at line **1008** of file [Filter.hpp](#).

Referenced by [getCount\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::Pipeline Member List

This is the complete list of members for **ob::Pipeline**, including all inherited members.

**disableFrameSync()** const  
**enableFrameSync()** const  
**FrameSetCallback** typedef  
**frameSetCallback**(ob\_frame\_t \*frameSet, void \*userData)  
**getCalibrationParam**(std::shared\_ptr< Config > config)  
**getCameraParam()**  
**getCameraParamWithProfile**(uint32\_t colorWidth, uint32\_t colorHeight, uint32\_t depthWidth, uint32\_t depth)  
**getConfig()** const  
**getD2CDepthProfileList**(std::shared\_ptr< StreamProfile > colorProfile, OBAlignMode alignMode)  
**getDevice()** const  
**getStreamProfileList**(OBSensorType sensorType) const  
**Pipeline()**  
**Pipeline**(std::shared\_ptr< Device > device)  
**start**(std::shared\_ptr< Config > config=nullptr) const  
**start**(std::shared\_ptr< Config > config, FrameSetCallback callback)  
**stop()** const  
**waitForFrames**(uint32\_t timeoutMs=1000) const  
**waitForFrameset**(uint32\_t timeoutMs=1000) const  
**~Pipeline()** noexcept

# ob::Pipeline Class Reference

```
#include <Pipeline.hpp>
```

## Public Types

```
typedef std::function< void(std::shared_ptr< FrameSet > frame)> FrameSetCallback
```

**FrameSetCallback** is a callback function type for frameset data arrival.

## Public Member Functions

```
Pipeline()
```

**Pipeline** is a high-level interface for applications, algorithms related RGBD data streams. **Pipeline** can provide alignment inside and synchronized **FrameSet**. **Pipeline()** no parameter version, which opens the first device in the list of devices connected to the OS by default. If the application has obtained the device through the **DeviceList**, opening the **Pipeline()** at this time will throw an exception that the device has been created.

```
Pipeline(std::shared_ptr< Device > device)
```

**Pipeline(std::shared\_ptr< Device > device )** Function for multi-device operations. Multiple devices need to be obtained through **DeviceList**, and the device and pipeline are bound through this interface.

```
~Pipeline() noexcept
```

Destroy the pipeline object.

```
void start(std::shared_ptr< Config > config=nullptr) const
```

Start the pipeline with configuration parameters.

```
void start(std::shared_ptr< Config > config, FrameSetCallback  
callback)
```

Start the pipeline and set the frameset data callback.

```
void stop() const
```

Stop the pipeline.

```
std::shared_ptr< Config > getConfig() const
```

Get the pipeline configuration parameters.

```
std::shared_ptr< FrameSet > waitForFrameset(uint32_t timeoutMs=1000) const
```

Wait for frameset.

```
std::shared_ptr< Device > getDevice() const
```

```

        Get the device object.

std::shared_ptr<StreamProfileList> getStreamProfileList (OBSensorType sensorType) const
        Get the stream profile of the specified sensor.

std::shared_ptr<StreamProfileList> getD2CDepthProfileList (std::shared_ptr<StreamProfile>
    colorProfile, OBAlignMode alignMode)
        Get the stream profile list of supported depth-to-color alignments.

void enableFrameSync () const
        Turn on frame synchronization.

void disableFrameSync () const
        Turn off frame synchronization.

OBCameraParam getCameraParam ()
OBCameraParam getCameraParamWithProfile (uint32_t colorWidth, uint32_t
    colorHeight, uint32_t depthWidth, uint32_t depthHeight)
OBCalibrationParam getCalibrationParam (std::shared_ptr<Config> config)

std::shared_ptr<FrameSet> waitForFrames (uint32_t timeoutMs=1000) const

```

## Static Public Member Functions

static void **frameSetCallback** (ob\_frame\_t \*frameSet, void \*userData)

## Detailed Description

Definition at line **249** of file [Pipeline.hpp](#).

## Member Typedef Documentation

### ◆ FrameSetCallback

typedef std::function<void(std::shared\_ptr<**FrameSet**> frame)> **ob::Pipeline::FrameSetCallback**

**FrameSetCallback** is a callback function type for frameset data arrival.

### Parameters

**frame** The returned frameset data

Definition at line **256** of file [Pipeline.hpp](#).

## Constructor & Destructor Documentation

### ◆ Pipeline() [1/2]

ob::Pipeline::Pipeline ( )

inline

**Pipeline** is a high-level interface for applications, algorithms related RGBD data streams. **Pipeline** can provide alignment inside and synchronized **FrameSet**. **Pipeline()** no parameter version, which opens the first device in the list of devices connected to the OS by default. If the application has obtained the device through the **DeviceList**, opening the **Pipeline()** at this time will throw an exception that the device has been created.

Definition at line **268** of file **Pipeline.hpp**.

Referenced by **frameSetCallback()**.

### ◆ Pipeline() [2/2]

ob::Pipeline::Pipeline ( std::shared\_ptr<**Device** > device )

inline explicit

**Pipeline(std::shared\_ptr<Device> device)** Function for multi-device operations. Multiple devices need to be obtained through **DeviceList**, and the device and pipeline are bound through this interface.

Definition at line **279** of file **Pipeline.hpp**.

### ◆ ~Pipeline()

ob::Pipeline::~Pipeline ( )

inline noexcept

Destroy the pipeline object.

Definition at line **288** of file **Pipeline.hpp**.

## Member Function Documentation

### ◆ start() [1/2]

```
void ob::Pipeline::start ( std::shared_ptr<Config> config = nullptr ) const
```

inline

Start the pipeline with configuration parameters.

#### Parameters

**config** The parameter configuration of the pipeline

Definition at line [299](#) of file [Pipeline.hpp](#).

#### ◆ **start()** [2/2]

```
void ob::Pipeline::start ( std::shared_ptr<Config> config,  
                           FrameSetCallback           callback )
```

inline

Start the pipeline and set the frameset data callback.

#### Parameters

**config** The configuration of the pipeline

**callback** The callback to be triggered when all frame data in the frameset arrives

Definition at line [312](#) of file [Pipeline.hpp](#).

#### ◆ **frameSetCallback()**

```
void ob::Pipeline::frameSetCallback ( ob_frame_t * frameSet,  
                                      void *           userData )
```

inline static

Definition at line [319](#) of file [Pipeline.hpp](#).

Referenced by [\*\*start\(\)\*\*](#).

#### ◆ **stop()**

```
void ob::Pipeline::stop ( ) const
```

inline

Stop the pipeline.

Definition at line [327](#) of file [Pipeline.hpp](#).

## ◆ getConfig()

std::shared\_ptr<**Config**> ob::Pipeline::getConfig ( ) const

inline

Get the pipeline configuration parameters.

Returns the default configuration if the user has not configured it

### Returns

std::shared\_ptr<Config> The configured parameters

Definition at line [339](#) of file **Pipeline.hpp**.

## ◆ waitForFrameset()

std::shared\_ptr<**FrameSet**> ob::Pipeline::waitForFrameset ( uint32\_t timeoutMs = 1000 ) const

inline

Wait for frameset.

### Parameters

**timeoutMs** The waiting timeout in milliseconds

### Returns

std::shared\_ptr<FrameSet> The waiting frameset data

Definition at line [352](#) of file **Pipeline.hpp**.

Referenced by [waitForFrames\(\)](#).

## ◆ getDevice()

std::shared\_ptr<**Device**> ob::Pipeline::getDevice ( ) const

inline

Get the device object.

### Returns

std::shared\_ptr<Device> The device object

Definition at line [367](#) of file **Pipeline.hpp**.

## ◆ getStreamProfileList()

```
std::shared_ptr<StreamProfileList>
ob::Pipeline::getStreamProfileList
( OBSensorType sensorType ) const
```

inline

Get the stream profile of the specified sensor.

### Parameters

**sensorType** The type of sensor

### Returns

std::shared\_ptr<StreamProfileList> The stream profile list

Definition at line **380** of file [Pipeline.hpp](#).

## ◆ **getD2CDepthProfileList()**

```
std::shared_ptr<StreamProfileList>
ob::Pipeline::getD2CDepthProfileList
( std::shared_ptr<StreamProfile> colorProfile,
  OBAAlignMode alignMode ) inline
```

Get the stream profile list of supported depth-to-color alignments.

### Parameters

**colorProfile** The color stream profile, which is the target stream profile for the depth-to-color alignment.

**alignMode** The alignment mode.

### Attention

Currently, only ALIGN\_D2C\_HW\_MODE supported. For other align modes, please using the AlignFilter interface.

### Returns

std::shared\_ptr<StreamProfileList> The stream profile list of supported depth-to-color alignments.

Definition at line **397** of file [Pipeline.hpp](#).

## ◆ **enableFrameSync()**

```
void ob::Pipeline::enableFrameSync( ) const
```

inline

Turn on frame synchronization.

Definition at line [407](#) of file [Pipeline.hpp](#).

◆ [disableFrameSync\(\)](#)

```
void ob::Pipeline::disableFrameSync( ) const
```

inline

Turn off frame synchronization.

Definition at line [416](#) of file [Pipeline.hpp](#).

◆ [getCameraParam\(\)](#)

```
OBCameraParam ob::Pipeline::getCameraParam( )
```

inline

Definition at line [425](#) of file [Pipeline.hpp](#).

◆ [getCameraParamWithProfile\(\)](#)

```
OBCameraParam ob::Pipeline::getCameraParamWithProfile( uint32_t colorWidth,  
                                                       uint32_t colorHeight,  
                                                       uint32_t depthWidth,  
                                                       uint32_t depthHeight )
```

inline

Definition at line [432](#) of file [Pipeline.hpp](#).

◆ [getCalibrationParam\(\)](#)

```
OBCalibrationParam ob::Pipeline::getCalibrationParam( std::shared_ptr<Config> config )
```

inline

Definition at line [439](#) of file [Pipeline.hpp](#).

◆ [waitForFrames\(\)](#)

```
std::shared_ptr<FrameSet> ob::Pipeline::waitForFrames ( uint32_t timeoutMs = 1000 ) const inline
```

Definition at line **446** of file [\*\*Pipeline.hpp\*\*](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[\*\*Pipeline.hpp\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::PlaybackDevice Member List

This is the complete list of members for **ob::PlaybackDevice**, including all inherited members.

**Device**(ob\_device\_t \*impl)  
**Device**(Device &&other) noexcept  
**Device**(const Device &)=delete  
**DeviceFwUpdateCallback** typedef  
**deviceStateChangeCallback\_**  
**DeviceStateChangedCallback** typedef  
**deviceStateChangedCallback**(OBDeviceState state, const char \*message, void \*userData)  
**deviceUpgrade**(const char \*filePath, DeviceFwUpdateCallback callback, bool async=true)  
**deviceUpgradeFromData**(const uint8\_t \*firmwareData, uint32\_t firmwareDataSize, DeviceFwUpdateCallback callback)  
**enableGlobalTimestamp**(bool enable)  
**enableHeartbeat**(bool enable) const  
**exportSettingsAsPresetJsonData**(const char \*presetName, const uint8\_t \*\*data, uint32\_t \*dataSize)  
**exportSettingsAsPresetJsonFile**(const char \*filePath) const  
**fwUpdateCallback\_**  
**getAvailableFrameInterleaveList**() const  
**getAvailablePresetList**() const  
**getAvailablePresetResolutionConfigList**() const  
**getBoolProperty**(OBPropertyID propertyId) const  
**getBoolPropertyRange**(OBPropertyID propertyId) const  
**getCalibrationCameraParamList**()  
**getCurrentDepthModeName**()  
**getCurrentDepthWorkMode**() const  
**getCurrentPresetName**() const  
**getDepthWorkModeList**() const  
**getDeviceInfo**() const  
**getDeviceState**()  
**getDuration**() const  
**getExtensionInfo**(const std::string &infoKey) const  
**getFloatProperty**(OBPropertyID propertyId) const  
**getFloatPropertyRange**(OBPropertyID propertyId) const  
**getImpl**() const  
**getIntProperty**(OBPropertyID propertyId) const

```
getIntPropertyRange(OBPropertyID propertyId) const
getMultiDeviceSyncConfig() const
getPlaybackStatus() const
getPosition() const
getSensor(OBSensorType type) const
getSensorList() const
getStructuredData(OBPropertyID propertyId, uint8_t *data, uint32_t *dataSize) const
getSupportedMultiDeviceSyncModeBitmap() const
getSupportedProperty(uint32_t index) const
getSupportedPropertyCount() const
getTimestampResetConfig() const
impl_
isExtensionInfoExist(const std::string &infoKey) const
isFrameInterleaveSupported() const
isGlobalTimestampSupported() const
isPropertySupported(OBPropertyID propertyId, OBPermissionType permission) const
loadDepthFilterConfig(const char *filePath)
loadFrameInterleave(const char *frameInterleaveName) const
loadPreset(const char *presetName) const
loadPresetFromJsonData(const char *presetName, const uint8_t *data, uint32_t size)
loadPresetFromFile(const char *filePath) const
operator=(PlaybackDevice &&other) noexcept
operator=(const PlaybackDevice &)=delete
ob::Device::operator=(Device &&other) noexcept
ob::Device::operator=(const Device &)=delete
pause()
PlaybackDevice(const std::string &file)
PlaybackDevice(PlaybackDevice &&other) noexcept
PlaybackDevice(const PlaybackDevice &)=delete
readCustomerData(void *data, uint32_t *dataSize)
reboot() const
reboot(uint32_t delayMs) const
resume()
seek(const int64_t timestamp)
sendAndReceiveData(const uint8_t *sendData, uint32_t sendDataSize, uint8_t *receiveData, uint32_t *receiveDataSize) const
setBoolProperty(OBPropertyID propertyId, bool value) const
```

**setDeviceStateChangedCallback**(DeviceStateChangedCallback callback)  
**setFloatProperty**(OBPropertyID propertyId, float value) const  
**setIntProperty**(OBPropertyID propertyId, int32\_t value) const  
**setMultiDeviceSyncConfig**(const OBMultiDeviceSyncConfig &config) const  
**setPlaybackRate**(const float rate)  
**setPlaybackStatusChangeCallback**(PlaybackStatusChangeCallback callback)  
**setStructuredData**(OBPropertyID propertyId, const uint8\_t \*data, uint32\_t dataSize) const  
**setTimestampResetConfig**(const OBDeviceTimestampResetConfig &config) const  
**switchDepthWorkMode**(const OBDepthWorkMode &workMode) const  
**switchDepthWorkMode**(const char \*modeName) const  
**timerSyncWithHost**() const  
**timestampReset**() const  
**triggerCapture**() const  
**updateFirmware**(const char \*filePath, DeviceFwUpdateCallback callback, bool async=true)  
**updateFirmwareFromData**(const uint8\_t \*firmwareData, uint32\_t firmwareDataSize, DeviceFwUpdateCallback callback)  
**updateOptionalDepthPresets**(const char filePathList[][OB\_PATH\_MAX], uint8\_t pathCount, DeviceFwUpdateCallback callback)  
**writeCustomerData**(const void \*data, uint32\_t dataSize)  
~**Device**() noexcept  
~**PlaybackDevice**() noexcept override=default

# ob::PlaybackDevice Class Reference

```
#include <RecordPlayback.hpp>
```

Inheritance diagram for ob::PlaybackDevice:

## Public Member Functions

```
    PlaybackDevice (const std::string &file)
    virtual ~PlaybackDevice () noexcept override=default
        PlaybackDevice (PlaybackDevice &&other) noexcept
PlaybackDevice & operator= (PlaybackDevice &&other) noexcept
            PlaybackDevice (const PlaybackDevice &)=delete
PlaybackDevice & operator= (const PlaybackDevice &)=delete
                void pause ()
                    Pause the streaming data from the playback device.
                void resume ()
                    Resume the streaming data from the playback device.
                void seek (const int64_t timestamp)
                    Seek to a specific timestamp when playing back a recording.
                void setPlaybackRate (const float rate)
                    Set the playback rate of the playback device.
                void setPlaybackStatusChangeCallback (PlaybackStatusChangeCallback callback)
                    Set a callback function to be called when the playback status changes.
OBPlaybackStatus getPlaybackStatus () const
                Get the current playback status of the playback device.
                uint64_t getPosition () const
                    Get the current position of the playback device.
                uint64_t getDuration () const
                    Get the duration of the playback device.
```

Public Member Functions inherited from **ob::Device**

## Additional Inherited Members

Public Types inherited from **ob::Device**

Static Public Member Functions inherited from **ob::Device**

Protected Attributes inherited from **ob::Device**

## Detailed Description

Definition at line [71](#) of file [RecordPlayback.hpp](#).

## Constructor & Destructor Documentation

### ◆ PlaybackDevice() [1/3]

ob::PlaybackDevice::PlaybackDevice ( const std::string & file )

inline explicit

Definition at line [73](#) of file [RecordPlayback.hpp](#).

Referenced by [operator=\(\)](#), [operator=\(\)](#), [PlaybackDevice\(\)](#), and [PlaybackDevice\(\)](#).

### ◆ ~PlaybackDevice()

virtual ob::PlaybackDevice::~PlaybackDevice ( )

override virtual default noexcept

### ◆ PlaybackDevice() [2/3]

ob::PlaybackDevice::PlaybackDevice ( [PlaybackDevice](#) && other )

inline noexcept

Definition at line [81](#) of file [RecordPlayback.hpp](#).

### ◆ PlaybackDevice() [3/3]

ob::PlaybackDevice::PlaybackDevice ( const [PlaybackDevice](#) & )

delete

## Member Function Documentation

### ◆ operator=() [1/2]

[PlaybackDevice](#) & ob::PlaybackDevice::operator= ( [PlaybackDevice](#) && other )

inline noexcept

Definition at line [83](#) of file [RecordPlayback.hpp](#).

◆ **operator=()** [2/2]

**PlaybackDevice** & ob::PlaybackDevice::operator= ( const **PlaybackDevice** & )

delete

◆ **pause()**

void ob::PlaybackDevice::pause ( )

inline

Pause the streaming data from the playback device.

Definition at line **95** of file **RecordPlayback.hpp**.

◆ **resume()**

void ob::PlaybackDevice::resume ( )

inline

Resume the streaming data from the playback device.

Definition at line **104** of file **RecordPlayback.hpp**.

◆ **seek()**

void ob::PlaybackDevice::seek ( const int64\_t timestamp )

inline

Seek to a specific timestamp when playing back a recording.

**Parameters**

[in] **timestamp** The timestamp to seek to, in milliseconds.

Definition at line **114** of file **RecordPlayback.hpp**.

◆ **setPlaybackRate()**

```
void ob::PlaybackDevice::setPlaybackRate ( const float rate )
```

inline

Set the playback rate of the playback device.

#### Parameters

[in] **rate** The playback rate to set.

Definition at line [124](#) of file [RecordPlayback.hpp](#).

#### ◆ [setPlaybackStatusChangeCallback\(\)](#)

```
void ob::PlaybackDevice::setPlaybackStatusChangeCallback ( PlaybackStatusChangeCallback callback )
```

inline

Set a callback function to be called when the playback status changes.

#### Parameters

[in] **callback** The callback function to set.

Definition at line [134](#) of file [RecordPlayback.hpp](#).

#### ◆ [getPlaybackStatus\(\)](#)

```
OBPlaybackStatus ob::PlaybackDevice::getPlaybackStatus ( ) const
```

inline

Get the current playback status of the playback device.

#### Returns

The current playback status.

Definition at line [145](#) of file [RecordPlayback.hpp](#).

#### ◆ [getPosition\(\)](#)

```
uint64_t ob::PlaybackDevice::getPosition ( ) const
```

inline

Get the current position of the playback device.

#### Returns

The current position of the playback device, in milliseconds.

Definition at line [157](#) of file [RecordPlayback.hpp](#).

◆ **getDuration()**

uint64\_t ob::PlaybackDevice::getDuration ( ) const

inline

Get the duration of the playback device.

**Returns**

The duration of the playback device, in milliseconds.

Definition at line **169** of file [RecordPlayback.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[RecordPlayback.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::PointCloudFilter Member List

This is the complete list of members for **ob::PointCloudFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virtu
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>PointCloudFilter()</b>	<b>ob::PointCloudFilter</b>	inline
<b>process(std::shared_ptr&lt; const Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt; Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setCameraParam(OBCameraParam param)</b>	<b>ob::PointCloudFilter</b>	inline
<b>setColorDataNormalization(bool state)</b>	<b>ob::PointCloudFilter</b>	inline
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setCoordinateDataScaled(float factor)</b>	<b>ob::PointCloudFilter</b>	inline
<b>setCoordinateSystem(OBCoordinateSystemType type)</b>	<b>ob::PointCloudFilter</b>	inline
<b>setCreatePointFormat(OBFormat format)</b>	<b>ob::PointCloudFilter</b>	inline
<b> setFrameAlignState(bool state)</b>	<b>ob::PointCloudFilter</b>	inline
<b>setPositionDataScaled(float scale)</b>	<b>ob::PointCloudFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~PointCloudFilter() noexcept override=default</b>	<b>ob::PointCloudFilter</b>	virtual



# ob::PointCloudFilter Class Reference

The **PointCloudFilter** class is a subclass of **Filter** that generates point clouds. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::PointCloudFilter:

## Public Member Functions

**PointCloudFilter ()**

virtual ~**PointCloudFilter** () noexcept override=default

void **setCreatePointFormat (OBFormat format)**

Set the output pointcloud frame format.

void **setCoordinateDataScaled (float factor)**

Set the point cloud coordinate data zoom factor.

void **setColorDataNormalization (bool state)**

Set point cloud color data normalization.

void **setCoordinateSystem (OBCoordinateSystemType type)**

Set the point cloud coordinate system.

void **setPositionDataScaled (float scale)**

void **setFrameAlignState (bool state)**

void **setCameraParam (OBCameraParam param)**

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

The **PointCloudFilter** class is a subclass of **Filter** that generates point clouds.

Definition at line **341** of file **Filter.hpp**.

## Constructor & Destructor Documentation

◆ **PointCloudFilter()**

ob::PointCloudFilter::PointCloudFilter ( )

inline

Definition at line [343](#) of file [Filter.hpp](#).

◆ **~PointCloudFilter()**

virtual ob::PointCloudFilter::~PointCloudFilter ( )

override virtual default noexcept

## Member Function Documentation

◆ **setCreatePointFormat()**

void ob::PointCloudFilter::setCreatePointFormat ( **OBFormat** format )

inline

Set the output pointcloud frame format.

### Parameters

**format** The point cloud frame format: OB\_FORMAT\_POINT or OB\_FORMAT\_RGB\_POINT

Definition at line [357](#) of file [Filter.hpp](#).

Referenced by [setCreatePointFormat\(\)](#).

◆ **setCoordinateDataScaled()**

```
void ob::PointCloudFilter::setCoordinateDataScaled ( float factor )
```

inline

Set the point cloud coordinate data zoom factor.

Calling this function to set the scale will change the point coordinate scaling factor of the output point cloud frame, The point coordinate scaling factor for the output point cloud frame can be obtained via [PointsFrame::getCoordinateValueScale](#) function.

### Parameters

**factor** The scale factor.

Definition at line [369](#) of file [Filter.hpp](#).

Referenced by [setPositionDataScaled\(\)](#).

### ◆ [setColorDataNormalization\(\)](#)

```
void ob::PointCloudFilter::setColorDataNormalization ( bool state )
```

inline

Set point cloud color data normalization.

If normalization is required, the output point cloud frame's color data will be normalized to the range [0, 1].

### Attention

This function only works for when create point format is set to OB\_FORMAT\_RGB\_POINT.

### Parameters

**state** Whether normalization is required.

Definition at line [381](#) of file [Filter.hpp](#).

### ◆ [setCoordinateSystem\(\)](#)

```
void ob::PointCloudFilter::setCoordinateSystem ( OBCoordinateSystemType type )
```

inline

Set the point cloud coordinate system.

### Parameters

**type** The coordinate system type.

Definition at line [390](#) of file [Filter.hpp](#).

◆ [setPositionDataScaled\(\)](#)

void ob::PointCloudFilter::setPositionDataScaled ( float scale )

inline

Definition at line [396](#) of file [Filter.hpp](#).

◆ [setFrameAlignState\(\)](#)

void ob::PointCloudFilter::setFrameAlignState ( bool state )

inline

Definition at line [401](#) of file [Filter.hpp](#).

◆ [setCameraParam\(\)](#)

void ob::PointCloudFilter::setCameraParam ( [OBCameraParam](#) param )

inline

Definition at line [405](#) of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::PointCloudHelper Member List

This is the complete list of members for **ob::PointCloudHelper**, including all inherited members.

**savePointCloudToPly**(const char \*fileName, std::shared\_ptr< ob::Frame > frame, bool saveBinary, bool useMes

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::PointCloudHelper Class Reference

```
#include <Utils.hpp>
```

## Static Public Member Functions

```
static bool savePointCloudToPly (const char *fileName, std::shared_ptr<ob::Frame> frame, bool  
saveBinary, bool useMesh, float meshThreshold)  
    save point cloud to ply file.
```

## Detailed Description

Definition at line 169 of file [Utils.hpp](#).

## Member Function Documentation

### ◆ savePointCloudToPly()

```
bool ob::PointCloudHelper::savePointCloudToPly ( const char * fileName,  
                                                std::shared_ptr<ob::Frame> frame,  
                                                bool saveBinary,  
                                                bool useMesh,  
                                                float meshThreshold )    inline static
```

save point cloud to ply file.

#### Parameters

[in] <b>fileName</b>	Point cloud save path
[in] <b>frame</b>	Point cloud frame
[in] <b>saveBinary</b>	Binary or textual,true: binary, false: textual
[in] <b>useMesh</b>	Save mesh or not, true: save as mesh, false: not save as mesh
[in] <b>meshThreshold</b>	Distance threshold for creating faces in point cloud,default value :50

#### Returns

bool save point cloud result

Definition at line 182 of file [Utils.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Utils.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::PointsFrame Member List

This is the complete list of members for **ob::PointsFrame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getCoordinateValueScale() const</b>	<b>ob::PointsFrame</b>	inline
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getHeight() const</b>	<b>ob::PointsFrame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getWidth() const</b>	<b>ob::PointsFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	
<b>metadata() const</b>	<b>ob::Frame</b>	inline
<b>metadataSize() const</b>	<b>ob::Frame</b>	inline
<b>PointsFrame(const ob_frame *impl)</b>	<b>ob::PointsFrame</b>	inline explicit

<b>systemTimeStamp()</b> const	<b>ob::Frame</b>	inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~PointsFrame()</b> noexcept override=default	<b>ob::PointsFrame</b>	

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::PointsFrame Class Reference

Define the **PointsFrame** class, which inherits from the **Frame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::PointsFrame:

## Public Member Functions

**PointsFrame** (const **ob\_frame** \*impl)

Construct a new **PointsFrame** object with a given pointer to the internal frame object.

**~PointsFrame** () noexcept override=default

float **getCoordinateValueScale** () const

Get the point coordinate value scale of the points frame. The point position value of the points frame is multiplied by the scale to give a position value in millimeters. For example, if scale=0.1, the x-coordinate value of a point is x = 10000, which means that the actual x-coordinate value = x\*scale = 10000\*0.1 = 1000mm.

uint32\_t **getWidth** () const

Get the width of the frame.

uint32\_t **getHeight** () const

Get the height of the frame.

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **PointsFrame** class, which inherits from the **Frame** class.

The **PointsFrame** class is used to obtain pointcloud data and point cloud information.

### Note

The pointcloud data format can be obtained from the **Frame::getFormat()** function. Witch can be one of the following formats:

- OB\_FORMAT\_POINT: 32-bit float format with 3D point coordinates (x, y, z), **OBPoint**
- OB\_FORMAT\_RGB\_POINT: 32-bit float format with 3D point coordinates (x, y, z) and point colors (r, g, b) **OBCColorPoint**

Definition at line **586** of file [Frame.hpp](#).

## Constructor & Destructor Documentation

### ◆ PointsFrame()

ob::PointsFrame::PointsFrame ( const **ob\_frame** \* impl )

inline explicit

Construct a new **PointsFrame** object with a given pointer to the internal frame object.

#### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

Please use the **FrameFactory** to create a **Frame** object.

#### Parameters

**impl** The pointer to the internal frame object.

Definition at line **599** of file [Frame.hpp](#).

### ◆ ~PointsFrame()

ob::PointsFrame::~PointsFrame ( )

override default noexcept

## Member Function Documentation

### ◆ getCoordinateValueScale()

```
float ob::PointsFrame::getCoordinateValueScale ( ) const
```

inline

Get the point coordinate value scale of the points frame. The point position value of the points frame is multiplied by the scale to give a position value in millimeters. For example, if scale=0.1, the x-coordinate value of a point is x = 10000, which means that the actual x-coordinate value = x\*scale = 10000\*0.1 = 1000mm.

### Returns

float The coordinate value scale.

Definition at line **610** of file [Frame.hpp](#).

Referenced by [getCoordinateValueScale\(\)](#).

### ◆ [getWidth\(\)](#)

```
uint32_t ob::PointsFrame::getWidth ( ) const
```

inline

Get the width of the frame.

### Returns

uint32\_t The width of the frame.

Definition at line **623** of file [Frame.hpp](#).

### ◆ [getHeight\(\)](#)

```
uint32_t ob::PointsFrame::getHeight ( ) const
```

inline

Get the height of the frame.

### Returns

uint32\_t The height of the frame.

Definition at line **637** of file [Frame.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)

## ob::PresetResolutionConfigeList Member List

This is the complete list of members for **ob::PresetResolutionConfigeList**, including all inherited members.

<b>getCount()</b>	<b>ob::PresetResolutionConfigeList</b>
<b>getPresetResolutionRatioConfig(uint32_t index)</b>	<b>ob::PresetResolutionConfigeList</b>
<b>PresetResolutionConfigeList(ob_preset_resolution_config_list_t *impl)</b>	<b>ob::PresetResolutionConfigeList</b>
<b>~PresetResolutionConfigeList() noexcept</b>	<b>ob::PresetResolutionConfigeList</b>

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::PresetResolutionConfigList Class Reference

Class representing a list of device **Frame** Interleave. [More...](#)

```
#include <Device.hpp>
```

## Public Member Functions

**PresetResolutionConfigList** (ob\_preset\_resolution\_config\_list\_t \*impl)

**~PresetResolutionConfigList** () noexcept

**uint32\_t getCount()**

Get the number of device preset resolution ratio in the list.

**OBPresetResolutionConfig getPresetResolutionRatioConfig** (uint32\_t index)

## Detailed Description

Class representing a list of device **Frame** Interleave.

Definition at line [1521](#) of file [Device.hpp](#).

## Constructor & Destructor Documentation

◆ **PresetResolutionConfigList()**

```
ob::PresetResolutionConfigList::PresetResolutionConfigList ( ob_preset_resolution_config_list_t * impl )    inline
```

Definition at line [1526](#) of file [Device.hpp](#).

◆ **~PresetResolutionConfigList()**

```
ob::PresetResolutionConfigList::~PresetResolutionConfigList ( )                                inline noexcept
```

Definition at line [1527](#) of file [Device.hpp](#).

## Member Function Documentation

◆ **getCount()**

uint32\_t ob::PresetResolutionConfigList::getCount ( )

inline

Get the number of device preset resolution ratio in the list.

Definition at line **1536** of file **Device.hpp**.

◆ **getPresetResolutionRatioConfig()**

**OBPresetResolutionConfig**

ob::PresetResolutionConfigList::getPresetResolutionRatioConfig

( uint32\_t index ) inline

Definition at line **1548** of file **Device.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Device.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::RecordDevice Member List

This is the complete list of members for **ob::RecordDevice**, including all inherited members.

<b>operator=(</b> RecordDevice &&other) noexcept	<b>ob::R</b>
<b>operator=(</b> const RecordDevice &)=delete	<b>ob::R</b>
<b>pause()</b>	<b>ob::R</b>
<b>RecordDevice</b> (std::shared_ptr< Device > device, const std::string &file, bool compressionEnabled=true)	<b>ob::R</b>
<b>RecordDevice</b> (RecordDevice &&other) noexcept	<b>ob::R</b>
<b>RecordDevice</b> (const RecordDevice &)=delete	<b>ob::R</b>
<b>resume()</b>	<b>ob::R</b>
<b>~RecordDevice()</b> noexcept	<b>ob::R</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::RecordDevice Class Reference

```
#include <RecordPlayback.hpp>
```

## Public Member Functions

```
    RecordDevice (std::shared_ptr<Device> device, const std::string &file, bool  
        compressionEnabled=true)  
    virtual ~RecordDevice () noexcept  
    RecordDevice (RecordDevice &&other) noexcept  
RecordDevice & operator= (RecordDevice &&other) noexcept  
    RecordDevice (const RecordDevice &)=delete  
RecordDevice & operator= (const RecordDevice &)=delete  
    void pause ()  
    void resume ()
```

## Detailed Description

Definition at line [21](#) of file [RecordPlayback.hpp](#).

## Constructor & Destructor Documentation

### ◆ **RecordDevice()** [1/3]

```
ob::RecordDevice::RecordDevice ( std::shared_ptr<Device> device,  
                                const std::string &           file,  
                                bool                      compressionEnabled = true )           inline explicit
```

Definition at line [26](#) of file [RecordPlayback.hpp](#).

Referenced by [operator=\(\)](#), [operator=\(\)](#), [RecordDevice\(\)](#), and [RecordDevice\(\)](#).

### ◆ ~**RecordDevice()**

virtual ob::RecordDevice::~RecordDevice ( ) inline virtual noexcept

Definition at line [32](#) of file [RecordPlayback.hpp](#).

◆ RecordDevice() [2/3]

ob::RecordDevice::RecordDevice ( **RecordDevice** && other ) inline noexcept

Definition at line [38](#) of file [RecordPlayback.hpp](#).

◆ RecordDevice() [3/3]

ob::RecordDevice::RecordDevice ( const **RecordDevice** & ) delete

## Member Function Documentation

◆ operator=() [1/2]

**RecordDevice** & ob::RecordDevice::operator= ( **RecordDevice** && other ) inline noexcept

Definition at line [45](#) of file [RecordPlayback.hpp](#).

◆ operator=() [2/2]

**RecordDevice** & ob::RecordDevice::operator= ( const **RecordDevice** & ) delete

◆ pause()

void ob::RecordDevice::pause ( ) inline

Definition at line [58](#) of file [RecordPlayback.hpp](#).

◆ resume()

```
void ob::RecordDevice::resume( )
```

inline

Definition at line **64** of file **RecordPlayback.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**RecordPlayback.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::Sensor Member List

This is the complete list of members for **ob::Sensor**, including all inherited members.

<b>callback_</b>	<b>ob::Sensor</b> protected
<b>createRecommendedFilters()</b> const	<b>ob::Sensor</b> inline
<b>FrameCallback</b> typedef	<b>ob::Sensor</b>
<b>getRecommendedFilters()</b> const	<b>ob::Sensor</b> inline
<b>getStreamProfileList()</b> const	<b>ob::Sensor</b> inline
<b>getType()</b> const	<b>ob::Sensor</b> inline
<b>impl_</b>	<b>ob::Sensor</b> protected
<b>operator=(Sensor &amp;&amp;sensor)</b> noexcept	<b>ob::Sensor</b> inline
<b>operator=(const Sensor &amp;sensor)=delete</b>	<b>ob::Sensor</b>
<b>Sensor(ob_sensor_t *impl)</b>	<b>ob::Sensor</b> inline explicit
<b>Sensor(Sensor &amp;&amp;sensor)</b> noexcept	<b>ob::Sensor</b> inline
<b>Sensor(const Sensor &amp;sensor)=delete</b>	<b>ob::Sensor</b>
<b>start(std::shared_ptr&lt; StreamProfile &gt; streamProfile, FrameCallback callback)</b>	<b>ob::Sensor</b> inline
<b>stop()</b> const	<b>ob::Sensor</b> inline
<b>switchProfile(std::shared_ptr&lt; StreamProfile &gt; streamProfile)</b>	<b>ob::Sensor</b> inline
<b>type()</b> const	<b>ob::Sensor</b> inline
<b>~Sensor()</b> noexcept	<b>ob::Sensor</b> inline virtual

# ob::Sensor Class Reference

```
#include <Sensor.hpp>
```

## Public Types

```
typedef std::function< void(std::shared_ptr<Frame > frame)> FrameCallback  
Callback function for frame data.
```

## Public Member Functions

```
Sensor (ob_sensor_t *impl)  
Sensor (Sensor &&sensor) noexcept  
Sensor & operator= (Sensor &&sensor) noexcept  
Sensor (const Sensor &sensor)=delete  
Sensor & operator= (const Sensor &sensor)=delete  
virtual ~Sensor () noexcept  
OBSensorType getType () const  
Get the sensor type.  
std::shared_ptr<StreamProfileList> getStreamProfileList () const  
Get the list of stream profiles.  
std::vector< std::shared_ptr<Filter > > createRecommendedFilters () const  
Create a list of recommended filters for the sensor.  
void start (std::shared_ptr<StreamProfile> streamProfile,  
FrameCallback callback)  
Open a frame data stream and set up a callback.  
void stop () const  
Stop the stream.  
void switchProfile (std::shared_ptr<StreamProfile> streamProfile)  
Dynamically switch resolutions.  
OBSensorType type () const  
std::vector< std::shared_ptr<Filter > > getRecommendedFilters () const
```

## Protected Attributes

```
ob_sensor_t * impl_  
FrameCallback callback_
```

## Detailed Description

Definition at line **24** of file [Sensor.hpp](#).

## Member Typedef Documentation

### ◆ FrameCallback

```
typedef std::function<void(std::shared_ptr<Frame> frame)> ob::Sensor::FrameCallback
```

Callback function for frame data.

## Parameters

**frame** The frame data.

Definition at line **31** of file [Sensor.hpp](#).

## Constructor & Destructor Documentation

### ◆ Sensor() [1/3]

```
ob::Sensor::Sensor ( ob_sensor_t * impl )inline explicit
```

Definition at line **38** of file [Sensor.hpp](#).

Referenced by [operator=\(\)](#), [operator=\(\)](#), [Sensor\(\)](#), and [Sensor\(\)](#).

### ◆ Sensor() [2/3]

```
ob::Sensor::Sensor ( Sensor && sensor )inline noexcept
```

Definition at line **40** of file [Sensor.hpp](#).

### ◆ Sensor() [3/3]

```
ob::Sensor::Sensor ( const Sensor & sensor )delete
```

◆ ~Sensor()

virtual ob::Sensor::~Sensor( )

inline virtual noexcept

Definition at line **58** of file [Sensor.hpp](#).

## Member Function Documentation

◆ operator=() [1/2]

**Sensor** & ob::Sensor::operator= ( **Sensor** && sensor )

inline noexcept

Definition at line **44** of file [Sensor.hpp](#).

◆ operator=() [2/2]

**Sensor** & ob::Sensor::operator= ( const **Sensor** & sensor )

delete

◆ getType()

**OBSensorType** ob::Sensor::getType( ) const

inline

Get the sensor type.

### Returns

**OBSensorType** The sensor type.

Definition at line **69** of file [Sensor.hpp](#).

Referenced by [type\(\)](#).

◆ getStreamProfileList()

```
std::shared_ptr<StreamProfileList> ob::Sensor::getStreamProfileList ( ) const
```

inline

Get the list of stream profiles.

#### Returns

```
std::shared_ptr<StreamProfileList> The stream profile list.
```

Definition at line **81** of file [Sensor.hpp](#).

#### ◆ **createRecommendedFilters()**

```
std::vector< std::shared_ptr<Filter>> ob::Sensor::createRecommendedFilters ( ) const
```

inline

Create a list of recommended filters for the sensor.

#### Returns

**OBFilterList** list of frame processing block

Definition at line **93** of file [Sensor.hpp](#).

Referenced by [getRecommendedFilters\(\)](#).

#### ◆ **start()**

```
void ob::Sensor::start ( std::shared_ptr<StreamProfile> streamProfile,
```

**FrameCallback** callback )

inline

Open a frame data stream and set up a callback.

#### Parameters

**streamProfile** The stream configuration.

**callback** The callback to set when frame data arrives.

Definition at line **116** of file [Sensor.hpp](#).

#### ◆ **stop()**

```
void ob::Sensor::stop ( ) const
```

inline

Stop the stream.

Definition at line [126](#) of file [Sensor.hpp](#).

◆ [switchProfile\(\)](#)

```
void ob::Sensor::switchProfile ( std::shared_ptr<StreamProfile> streamProfile )
```

inline

Dynamically switch resolutions.

**Parameters**

**streamProfile** The resolution to switch to.

Definition at line [137](#) of file [Sensor.hpp](#).

◆ [type\(\)](#)

```
OBSensorType ob::Sensor::type ( ) const
```

inline

Definition at line [151](#) of file [Sensor.hpp](#).

Referenced by [get\\_type\(\)](#).

◆ [getRecommendedFilters\(\)](#)

```
std::vector< std::shared_ptr<Filter>> ob::Sensor::getRecommendedFilters ( ) const
```

inline

Definition at line [155](#) of file [Sensor.hpp](#).

## Member Data Documentation

◆ [impl\\_](#)

ob\_sensor\_t\* ob::Sensor::impl\_

protected

Definition at line [34](#) of file [Sensor.hpp](#).

Referenced by [createRecommendedFilters\(\)](#), [getStreamProfileList\(\)](#), [getType\(\)](#), [operator=\(\)](#), [Sensor\(\)](#), [start\(\)](#), [stop\(\)](#), [switchProfile\(\)](#), and [~Sensor\(\)](#).

◆ [callback\\_](#)

**FrameCallback** ob::Sensor::callback\_

protected

Definition at line [35](#) of file [Sensor.hpp](#).

Referenced by [start\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Sensor.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::SensorList Member List

This is the complete list of members for **ob::SensorList**, including all inherited members.

<b>count()</b> const	<b>ob::SensorList</b> inline
<b>getCount()</b> const	<b>ob::SensorList</b> inline
<b>getSensor(uint32_t index)</b> const	<b>ob::SensorList</b> inline
<b>getSensor(OBSensorType sensorType)</b> const	<b>ob::SensorList</b> inline
<b>getSensorType(uint32_t index)</b> const	<b>ob::SensorList</b> inline
<b>SensorList(ob_sensor_list_t *impl)</b>	<b>ob::SensorList</b> inline explicit
<b>type(uint32_t index)</b> const	<b>ob::SensorList</b> inline
<b>~SensorList()</b> noexcept	<b>ob::SensorList</b> inline

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::SensorList Class Reference

```
#include <Sensor.hpp>
```

## Public Member Functions

```
SensorList (ob_sensor_list_t *impl)  
~SensorList () noexcept  
uint32_t getCount () const  
    Get the number of sensors.  
OBSensorType getSensorType (uint32_t index) const  
    Get the type of the specified sensor.  
std::shared_ptr<Sensor> getSensor (uint32_t index) const  
    Get a sensor by index number.  
std::shared_ptr<Sensor> getSensor (OBSensorType sensorType) const  
    Get a sensor by sensor type.  
uint32_t count () const  
OBSensorType type (uint32_t index) const
```

## Detailed Description

Definition at line **160** of file [Sensor.hpp](#).

## Constructor & Destructor Documentation

### ◆ SensorList()

```
ob::SensorList::SensorList ( ob_sensor_list_t * impl )  
    inline explicit
```

Definition at line **165** of file [Sensor.hpp](#).

### ◆ ~SensorList()

```
ob::SensorList::~SensorList ( )
```

inline noexcept

Definition at line [167](#) of file [Sensor.hpp](#).

## Member Function Documentation

### ◆ getCount()

```
uint32_t ob::SensorList::getCount ( ) const
```

inline

Get the number of sensors.

#### Returns

`uint32_t` The number of sensors.

Definition at line [178](#) of file [Sensor.hpp](#).

Referenced by [count\(\)](#).

### ◆ getSensorType()

```
OBSensorType ob::SensorList::getSensorType ( uint32_t index ) const
```

inline

Get the type of the specified sensor.

#### Parameters

`index` The sensor index.

#### Returns

`OBSensorType` The sensor type.

Definition at line [191](#) of file [Sensor.hpp](#).

Referenced by [type\(\)](#).

### ◆ getSensor() [1/2]

```
std::shared_ptr<Sensor> ob::SensorList::getSensor ( uint32_t index ) const
```

inline

Get a sensor by index number.

#### Parameters

**index** The sensor index. The range is [0, count-1]. If the index exceeds the range, an exception will be thrown.

#### Returns

```
std::shared_ptr<Sensor> The sensor object.
```

Definition at line [204](#) of file [Sensor.hpp](#).

### ◆ [getSensor\(\)](#) [2/2]

```
std::shared_ptr<Sensor> ob::SensorList::getSensor ( OBSSensorType sensorType ) const
```

inline

Get a sensor by sensor type.

#### Parameters

**sensorType** The sensor type to obtain.

#### Returns

```
std::shared_ptr<Sensor> A sensor object. If the specified sensor type does not exist, it will return empty.
```

Definition at line [217](#) of file [Sensor.hpp](#).

### ◆ [count\(\)](#)

```
uint32_t ob::SensorList::count ( ) const
```

inline

Definition at line [226](#) of file [Sensor.hpp](#).

Referenced by [getCount\(\)](#).

### ◆ [type\(\)](#)

**OBSensorType** ob::SensorList::type ( uint32\_t index ) const

inline

Definition at line **230** of file **Sensor.hpp**.

Referenced by **getSensorType()**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Sensor.hpp**

Generated on for OrbbecSDK by **doxygen** 1.14.0

# ob::SequenceIdFilter Member List

This is the complete list of members for **ob::SequenceIdFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>getSelectSequenceId()</b>	<b>ob::SequenceIdFilter</b>	inline
<b>getSequenceIdList()</b>	<b>ob::SequenceIdFilter</b>	inline
<b>getSequenceIdListSize()</b>	<b>ob::SequenceIdFilter</b>	inline
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virt
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt;const Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt;Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>selectSequenceId(int sequence_id)</b>	<b>ob::SequenceIdFilter</b>	inline
<b>SequenceIdFilter()</b>	<b>ob::SequenceIdFilter</b>	inline
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~SequenceIdFilter() noexcept override</b>	<b>ob::SequenceIdFilter</b>	inline virtual

# ob::SequenceIdFilter Class Reference

Create **SequenceIdFilter** processing block. More...

```
#include <Filter.hpp>
```

Inheritance diagram for ob::SequenceIdFilter:

## Public Member Functions

```
    SequenceIdFilter()
    virtual ~SequenceIdFilter() noexcept override
    void selectSequenceId(int sequence_id)
        Set the sequenceId filter params.
    int getSelectSequenceId()
        Get the current sequence id.
    OBSequenceIdItem * getSequenceIdList()
    int getSequenceIdListSize()
        Get the sequenceId list size.
```

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

Create **SequenceIdFilter** processing block.

Definition at line **501** of file **Filter.hpp**.

## Constructor & Destructor Documentation

- ◆ **SequenceIdFilter()**

ob::SequenceIdFilter::SequenceIdFilter( ) inline

Definition at line [519](#) of file [Filter.hpp](#).

◆ ~SequenceIdFilter()

virtual ob::SequenceIdFilter::~SequenceIdFilter( ) inline override virtual noexcept

Definition at line [527](#) of file [Filter.hpp](#).

## Member Function Documentation

◆ selectSequenceId()

void ob::SequenceIdFilter::selectSequenceId ( int sequence\_id ) inline

Set the sequenceId filter params.

### Parameters

**sequence\_id** id to pass the filter.

Definition at line [539](#) of file [Filter.hpp](#).

◆ getSelectSequenceId()

int ob::SequenceIdFilter::getSelectSequenceId ( ) inline

Get the current sequence id.

### Returns

sequence id to pass the filter.

Definition at line [548](#) of file [Filter.hpp](#).

◆ getSequenceIdList()

**OBSequenceIdItem**\* ob::SequenceIdFilter::getSequenceIdList( )

inline

Definition at line **552** of file **Filter.hpp**.

◆ **getSequenceIdListSize()**

int ob::SequenceIdFilter::getSequenceIdListSize( )

inline

Get the sequenceId list size.

**Returns**

the size of sequenceId list.

Definition at line **561** of file **Filter.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

# ob::SpatialAdvancedFilter Member List

This is the complete list of members for **ob::SpatialAdvancedFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getAlphaRange()</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getDispDiffRange()</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>getFilterParams()</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getMagnitudeRange()</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>getRadiusRange()</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt; const Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt; Frame &gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setFilterParams(OBSpatialAdvancedFilterParams params)</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>SpatialAdvancedFilter(const std::string &amp;activationKey="")</b>	<b>ob::SpatialAdvancedFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~SpatialAdvancedFilter() noexcept override=default</b>	<b>ob::SpatialAdvancedFilter</b>	virtual



# ob::SpatialAdvancedFilter Class Reference

Spatial advanced filte smooths the image by calculating frame with alpha and delta settings alpha defines the weight of the current pixel for smoothing, delta defines the depth gradient below which the smoothing will occur as number of depth levels. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::SpatialAdvancedFilter:

## Public Member Functions

```
SpatialAdvancedFilter (const std::string &activationKey="")  
virtual ~SpatialAdvancedFilter () noexcept override=default  
OBFloatPropertyRange getAlphaRange ()  
Get the spatial advanced filter alpha range.  
OBUInt16PropertyRange getDispDiffRange ()  
Get the spatial advanced filter dispdiff range.  
OBUInt16PropertyRange getRadiusRange ()  
Get the spatial advanced filter radius range.  
OBIntPropertyRange getMagnitudeRange ()  
Get the spatial advanced filter magnitude range.  
OBSpatialAdvancedFilterParams getFilterParams ()  
Get the spatial advanced filter params.  
void setFilterParams (OBSpatialAdvancedFilterParams params)  
Set the spatial advanced filter params.
```

Public Member Functions inherited from [ob::Filter](#)

## Additional Inherited Members

Protected Member Functions inherited from [ob::Filter](#)

Protected Attributes inherited from [ob::Filter](#)

## Detailed Description

Spatial advanced filte smooths the image by calculating frame with alpha and delta settings alpha defines the weight of the current pixel for smoothing, delta defines the depth gradient below which the smoothing will occur as number of depth levels.

Definition at line **676** of file [Filter.hpp](#).

## Constructor & Destructor Documentation

### ◆ SpatialAdvancedFilter()

ob::SpatialAdvancedFilter::SpatialAdvancedFilter ( const std::string & activationKey = "" )

inline

Definition at line [678](#) of file [Filter.hpp](#).

### ◆ ~SpatialAdvancedFilter()

virtual ob::SpatialAdvancedFilter::~SpatialAdvancedFilter ( )

override virtual default noexcept

## Member Function Documentation

### ◆ getAlphaRange()

[OBFloatPropertyRange](#) ob::SpatialAdvancedFilter::getAlphaRange ( )

inline

Get the spatial advanced filter alpha range.

#### Returns

[OBFloatPropertyRange](#) the alpha value of property range.

Definition at line [692](#) of file [Filter.hpp](#).

Referenced by [getAlphaRange\(\)](#).

### ◆ getDispDiffRange()

[OBUInt16PropertyRange](#) ob::SpatialAdvancedFilter::getDispDiffRange ( )

inline

Get the spatial advanced filter dispdiff range.

#### Returns

[OBFloatPropertyRange](#) the dispdif value of property range.

Definition at line [709](#) of file [Filter.hpp](#).

◆ [getRadiusRange\(\)](#)

**OB UInt16PropertyRange** ob::SpatialAdvancedFilter::getRadiusRange ( )

inline

Get the spatial advanced filter radius range.

**Returns**

**OBFloatPropertyRange** the radius value of property range.

Definition at line [726](#) of file **Filter.hpp**.

◆ [getMagnitudeRange\(\)](#)

**OB IntPropertyRange** ob::SpatialAdvancedFilter::getMagnitudeRange ( )

inline

Get the spatial advanced filter magnitude range.

**Returns**

**OBFloatPropertyRange** the magnitude value of property range.

Definition at line [743](#) of file **Filter.hpp**.

◆ [getFilterParams\(\)](#)

**OB SpatialAdvancedFilterParams** ob::SpatialAdvancedFilter::getFilterParams ( )

inline

Get the spatial advanced filter params.

**Returns**

**OB SpatialAdvancedFilterParams**

Definition at line [760](#) of file **Filter.hpp**.

◆ [setFilterParams\(\)](#)

```
void ob::SpatialAdvancedFilter::setFilterParams ( OBSSpatialAdvancedFilterParams params )
```

inline

Set the spatial advanced filter params.

#### Parameters

params **OBSSpatialAdvancedFilterParams**.

Definition at line [774](#) of file **Filter.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::StreamProfile Member List

This is the complete list of members for **ob::StreamProfile**, including all inherited members.

<b>as()</b>	<b>ob::StreamProfile</b>
<b>as() const</b>	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo(std::shared_ptr&lt; StreamProfile &gt; target, const OBExtrinsic &amp;extrinsic)</b>	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo(const OBStreamType &amp;targetStreamType, const OBExtrinsic &amp;extrinsic)</b>	<b>ob::StreamProfile</b>
<b>format() const</b>	<b>ob::StreamProfile</b>
<b>getExtrinsicTo(std::shared_ptr&lt; StreamProfile &gt; target) const</b>	<b>ob::StreamProfile</b>
<b>getFormat() const</b>	<b>ob::StreamProfile</b>
<b>getImpl() const</b>	<b>ob::StreamProfile</b>
<b>getType() const</b>	<b>ob::StreamProfile</b>
<b>impl_</b>	<b>ob::StreamProfile</b>
<b>is() const</b>	<b>ob::StreamProfile</b>
<b>operator=(StreamProfile &amp;streamProfile)=delete</b>	<b>ob::StreamProfile</b>
<b>operator=(StreamProfile &amp;&amp;streamProfile) noexcept</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(StreamProfile &amp;streamProfile)=delete</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(StreamProfile &amp;&amp;streamProfile) noexcept</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(const ob_stream_profile_t *impl)</b>	<b>ob::StreamProfile</b>
<b>type() const</b>	<b>ob::StreamProfile</b>
<b>~StreamProfile() noexcept</b>	<b>ob::StreamProfile</b>

# ob::StreamProfile Class Reference

```
#include <StreamProfile.hpp>
```

Inheritance diagram for ob::StreamProfile:

## Public Member Functions

```
    StreamProfile (StreamProfile &streamProfile)=delete  
    StreamProfile & operator= (StreamProfile &streamProfile)=delete  
        StreamProfile (StreamProfile &&streamProfile) noexcept  
    StreamProfile & operator= (StreamProfile &&streamProfile) noexcept  
        virtual ~StreamProfile () noexcept  
const ob_stream_profile * getImpl () const  
    OBFormat getFormat () const  
        Get the format of the stream.  
    OBStreamType getType () const  
        Get the type of stream.  
    OBExtrinsic getExtrinsicTo (std::shared_ptr<StreamProfile> target) const  
        Get the extrinsic parameters from current stream profile to the given target  
        stream profile.  
    void bindExtrinsicTo (std::shared_ptr<StreamProfile> target, const  
        OBExtrinsic &extrinsic)  
        Set the extrinsic parameters from current stream profile to the given target  
        stream profile.  
    void bindExtrinsicTo (const OBStreamType &targetStreamType, const  
        OBExtrinsic &extrinsic)  
        Set the extrinsic parameters from current stream profile to the given target  
        stream type.  
  
template<typename T>  
    bool is () const  
        Check if frame object is compatible with the given type.  
  
template<typename T>  
    std::shared_ptr<T> as ()  
        Converts object type to target type.  
  
template<typename T>  
    std::shared_ptr<const T> as () const  
        Converts object type to target type (const version).  
    OBFormat format () const
```

**OBStreamType** type () const

## Protected Member Functions

**StreamProfile** (const ob\_stream\_profile\_t \*impl)

## Protected Attributes

const ob\_stream\_profile\_t \* **impl\_** = nullptr

## Detailed Description

Definition at line **18** of file **StreamProfile.hpp**.

## Constructor & Destructor Documentation

◆ **StreamProfile()** [1/3]

ob::StreamProfile::StreamProfile ( **StreamProfile** & streamProfile )

delete

Referenced by **ob::AccelStreamProfile::AccelStreamProfile()**,  
**ob::GyroStreamProfile::GyroStreamProfile()**, **operator=()**, **operator=()**, **StreamProfile()**,  
**StreamProfile()**, and **ob::VideoStreamProfile::VideoStreamProfile()**.

◆ **StreamProfile()** [2/3]

ob::StreamProfile::StreamProfile ( **StreamProfile** && streamProfile )

inline noexcept

Definition at line **26** of file **StreamProfile.hpp**.

◆ **~StreamProfile()**

virtual ob::StreamProfile::~StreamProfile ( )

inline virtual noexcept

Definition at line **41** of file **StreamProfile.hpp**.

◆ **StreamProfile()** [3/3]

```
ob::StreamProfile::StreamProfile ( const ob_stream_profile_t * impl )  
    inline explicit protected
```

Definition at line [159](#) of file [StreamProfile.hpp](#).

## Member Function Documentation

### ◆ operator=()

```
StreamProfile & ob::StreamProfile::operator= ( StreamProfile & streamProfile )  
    delete
```

### ◆ operator=()

```
StreamProfile & ob::StreamProfile::operator= ( StreamProfile && streamProfile )  
    inline noexcept
```

Definition at line [30](#) of file [StreamProfile.hpp](#).

### ◆ getImpl()

```
const ob_stream_profile * ob::StreamProfile::getImpl ( ) const  
    inline
```

Definition at line [49](#) of file [StreamProfile.hpp](#).

### ◆ getFormat()

```
OBFormat ob::StreamProfile::getFormat ( ) const  
    inline
```

Get the format of the stream.

#### Returns

**OBFormat** return the format of the stream.

Definition at line [58](#) of file [StreamProfile.hpp](#).

Referenced by [format\(\)](#).

### ◆ getType()

**OBStreamType** ob::StreamProfile::getType ( ) const

inline

Get the type of stream.

#### Returns

**OBStreamType** return the type of the stream.

Definition at line **70** of file **StreamProfile.hpp**.

Referenced by **is()**, and **type()**.

#### ◆ getExtrinsicTo()

**OBExtrinsic** ob::StreamProfile::getExtrinsicTo ( std::shared\_ptr<**StreamProfile**> target ) const

inline

Get the extrinsic parameters from current stream profile to the given target stream profile.

#### Returns

**OBExtrinsic** Return the extrinsic parameters.

Definition at line **82** of file **StreamProfile.hpp**.

#### ◆ bindExtrinsicTo() [1/2]

void ob::StreamProfile::bindExtrinsicTo ( std::shared\_ptr<**StreamProfile**> target,

const **OBExtrinsic** & extrinsic )

inline

Set the extrinsic parameters from current stream profile to the given target stream profile.

#### Template Parameters

**target** Target stream profile.

**extrinsic** The extrinsic.

Definition at line **95** of file **StreamProfile.hpp**.

#### ◆ bindExtrinsicTo() [2/2]

```
void ob::StreamProfile::bindExtrinsicTo ( const OBStreamType & targetStreamType,  
                                         const OBExtrinsic & extrinsic )  
                                         inline
```

Set the extrinsic parameters from current stream profile to the given target stream type.

### Template Parameters

**targetStreamType** Target stream type.  
**extrinsic** The extrinsic.

Definition at line [107](#) of file [StreamProfile.hpp](#).

### ◆ is()

```
template<typename T>  
bool ob::StreamProfile::is ( ) const
```

Check if frame object is compatible with the given type.

### Template Parameters

**T** Given type.

### Returns

bool return result.

Definition at line [381](#) of file [StreamProfile.hpp](#).

Referenced by [as\(\)](#), and [as\(\)](#).

### ◆ as() [1/2]

```
template<typename T>  
std::shared_ptr<T> ob::StreamProfile::as ( )  
                                         inline
```

Converts object type to target type.

### Template Parameters

**T** Target type.

### Returns

`std::shared_ptr<T>` Return the result. Throws an exception if conversion is not possible.

Definition at line [127](#) of file [StreamProfile.hpp](#).

◆ **as()** [2/2]

```
template<typename T>
std::shared_ptr<const T> ob::StreamProfile::as( ) const inline
```

Converts object type to target type (const version).

**Template Parameters**

**T**Target type.

**Returns**

`std::shared_ptr<T>` Return the result. Throws an exception if conversion is not possible.

Definition at line [141](#) of file **StreamProfile.hpp**.

◆ **format()**

```
OBFormat ob::StreamProfile::format( ) const inline
```

Definition at line [150](#) of file **StreamProfile.hpp**.

Referenced by **getFormat()**.

◆ **type()**

```
OBStreamType ob::StreamProfile::type( ) const inline
```

Definition at line [154](#) of file **StreamProfile.hpp**.

Referenced by **getType()**.

## Member Data Documentation

◆ **impl\_**

```
const ob_stream_profile_t* ob::StreamProfile::impl_ = nullptr
```

protected

Definition at line **20** of file [StreamProfile.hpp](#).

Referenced by [bindExtrinsicTo\(\)](#), [bindExtrinsicTo\(\)](#), [ob::VideoStreamProfile::getDistortion\(\)](#), [getExtrinsicTo\(\)](#), [getFormat\(\)](#), [ob::VideoStreamProfile::getFps\(\)](#), [ob::AccelStreamProfile::getFullScaleRange\(\)](#), [ob::GyroStreamProfile::getFullScaleRange\(\)](#), [ob::VideoStreamProfile::getHeight\(\)](#), [getImpl\(\)](#), [ob::AccelStreamProfile::getIntrinsic\(\)](#), [ob::GyroStreamProfile::getIntrinsic\(\)](#), [ob::VideoStreamProfile::getIntrinsic\(\)](#), [ob::AccelStreamProfile::getSampleRate\(\)](#), [ob::GyroStreamProfile::getSampleRate\(\)](#), [getType\(\)](#), [ob::VideoStreamProfile::getWidth\(\)](#), [operator=\(\)](#), [ob::VideoStreamProfile::setDistortion\(\)](#), [ob::VideoStreamProfile::setIntrinsic\(\)](#), [StreamProfile\(\)](#), and [~StreamProfile\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[StreamProfile.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::StreamProfileFactory Member List

This is the complete list of members for **ob::StreamProfileFactory**, including all inherited members.

**create**(const ob\_stream\_profile\_t \*impl) **ob::StreamProfileFactory** inline static

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::StreamProfileFactory Class Reference

```
#include <StreamProfile.hpp>
```

## Static Public Member Functions

```
static std::shared_ptr<StreamProfile> create (const ob_stream_profile_t *impl)
```

## Detailed Description

Definition at line **402** of file **StreamProfile.hpp**.

## Member Function Documentation

### ◆ **create()**

```
std::shared_ptr<StreamProfile>
ob::StreamProfileFactory::create
( const ob_stream_profile_t * impl )    inline static
```

Definition at line **404** of file **StreamProfile.hpp**.

Referenced by **ob::StreamProfileList::getAccelStreamProfile()**,  
**ob::StreamProfileList::getGyroStreamProfile()**, **ob::StreamProfileList::getProfile()**,  
**ob::Frame::getStreamProfile()**, and **ob::StreamProfileList::getVideoStreamProfile()**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**StreamProfile.hpp**

## ob::StreamProfileList Member List

This is the complete list of members for **ob::StreamProfileList**, including all inherited members.

**count()** const  
**getAccelStreamProfile(OBAccelFullScaleRange fullScaleRange, OBAccelSampleRate sampleRate)** const  
**getCount()** const  
**getGyroStreamProfile(OBGyroFullScaleRange fullScaleRange, OBGyroSampleRate sampleRate)** const  
**getProfile(uint32\_t index)** const  
**getVideoStreamProfile(int width=OB\_WIDTH\_ANY, int height=OB\_HEIGHT\_ANY, OBFormat format=OB\_Fimpl\_**  
**StreamProfileList(ob\_stream\_profile\_list\_t \*impl)**  
**~StreamProfileList()** noexcept

# ob::StreamProfileList Class Reference

```
#include <StreamProfile.hpp>
```

## Public Member Functions

```
StreamProfileList (ob_stream_profile_list_t *impl)  
~StreamProfileList () noexcept  
uint32_t getCount () const  
    Return the number of StreamProfile objects.  
std::shared_ptr<StreamProfile> getProfile (uint32_t index) const  
    Return the StreamProfile object at the specified index.  
std::shared_ptr<VideoStreamProfile> getVideoStreamProfile (int width=OB_WIDTH_ANY, int  
height=OB_HEIGHT_ANY, OBFormat  
format=OB_FORMAT_ANY, int fps=OB_FPS_ANY) const  
    Match the corresponding video stream profile based on the  
passed-in parameters. If multiple Match are found, the first one  
in the list is returned by default. Throws an exception if no  
matching profile is found.  
std::shared_ptr<AccelStreamProfile> getAccelStreamProfile (OBAccelFullScaleRange  
fullScaleRange, OBAccelSampleRate sampleRate) const  
    Match the corresponding accelerometer stream profile based on  
the passed-in parameters. If multiple Match are found, the first  
one in the list is returned by default. Throws an exception if no  
matching profile is found.  
std::shared_ptr<GyroStreamProfile> getGyroStreamProfile (OBGyroFullScaleRange  
fullScaleRange, OBGyroSampleRate sampleRate) const  
    Match the corresponding gyroscope stream profile based on the  
passed-in parameters. If multiple Match are found, the first one  
in the list is returned by default. Throws an exception if no  
matching profile is found.  
uint32_t count () const
```

## Protected Attributes

```
const ob_stream_profile_list_t * impl_
```

## Detailed Description

Definition at line [430](#) of file [StreamProfile.hpp](#).

## Constructor & Destructor Documentation

### ◆ StreamProfileList()

ob::StreamProfileList::StreamProfileList ( ob\_stream\_profile\_list\_t \* impl )

inline explicit

Definition at line [435](#) of file [StreamProfile.hpp](#).

### ◆ ~StreamProfileList()

ob::StreamProfileList::~StreamProfileList ( )

inline noexcept

Definition at line [436](#) of file [StreamProfile.hpp](#).

## Member Function Documentation

### ◆ getCount()

uint32\_t ob::StreamProfileList::getCount ( ) const

inline

Return the number of [StreamProfile](#) objects.

#### Returns

uint32\_t Return the number of [StreamProfile](#) objects.

Definition at line [447](#) of file [StreamProfile.hpp](#).

Referenced by [count\(\)](#).

### ◆ getProfile()

```
std::shared_ptr<StreamProfile> ob::StreamProfileList::getProfile ( uint32_t index ) const inline
```

Return the **StreamProfile** object at the specified index.

### Parameters

**index** The index of the **StreamProfile** object to be retrieved. Must be in the range [0, count-1].

Throws an exception if the index is out of range.

### Returns

```
std::shared_ptr<StreamProfile> Return the StreamProfile object.
```

Definition at line 460 of file **StreamProfile.hpp**.

### ◆ **getVideoStreamProfile()**

```
std::shared_ptr<VideoStreamProfile>
ob::StreamProfileList::getVideoStreamProfile ( int width = OB_WIDTH_ANY,
                                              int height = OB_HEIGHT_ANY,
                                              OBFormat format = OB_FORMAT_ANY,
                                              int fps = OB_FPS_ANY) const inline
```

Match the corresponding video stream profile based on the passed-in parameters. If multiple Match are found, the first one in the list is returned by default. Throws an exception if no matching profile is found.

### Parameters

**width** The width of the stream. Pass OB\_WIDTH\_ANY if no matching condition is required.

**height** The height of the stream. Pass OB\_HEIGHT\_ANY if no matching condition is required.

**format** The type of the stream. Pass OB\_FORMAT\_ANY if no matching condition is required.

**fps** The frame rate of the stream. Pass OB\_FPS\_ANY if no matching condition is required.

### Returns

```
std::shared_ptr<VideoStreamProfile> Return the matching resolution.
```

Definition at line 477 of file **StreamProfile.hpp**.

### ◆ **getAccelStreamProfile()**

```

std::shared_ptr<AccelStreamProfile>
ob::StreamProfileList::getAccelStreamProfile
( OBAccelFullScaleRange fullScaleRange,
OBAccelSampleRate sampleRate ) const inline

```

Match the corresponding accelerometer stream profile based on the passed-in parameters. If multiple Match are found, the first one in the list is returned by default. Throws an exception if no matching profile is found.

### Parameters

**fullScaleRange** The full scale range. Pass 0 if no matching condition is required.

**sampleRate** The sampling frequency. Pass 0 if no matching condition is required.

Definition at line [493](#) of file [StreamProfile.hpp](#).

- ◆ [getGyroStreamProfile\(\)](#)

```

std::shared_ptr<GyroStreamProfile>
ob::StreamProfileList::getGyroStreamProfile
( OBGyroFullScaleRange fullScaleRange,
OBGyroSampleRate sampleRate ) const inline

```

Match the corresponding gyroscope stream profile based on the passed-in parameters. If multiple Match are found, the first one in the list is returned by default. Throws an exception if no matching profile is found.

### Parameters

**fullScaleRange** The full scale range. Pass 0 if no matching condition is required.

**sampleRate** The sampling frequency. Pass 0 if no matching condition is required.

Definition at line [508](#) of file [StreamProfile.hpp](#).

- ◆ [count\(\)](#)

```

uint32_t ob::StreamProfileList::count( ) const
                                         inline

```

Definition at line [518](#) of file [StreamProfile.hpp](#).

Referenced by [getCount\(\)](#).

## Member Data Documentation

◆ **impl\_**

const ob\_stream\_profile\_list\_t\* ob::StreamProfileList::impl\_

protected

Definition at line [432](#) of file [StreamProfile.hpp](#).

Referenced by [getAccelStreamProfile\(\)](#), [getCount\(\)](#), [getGyroStreamProfile\(\)](#), [getProfile\(\)](#), [getVideoStreamProfile\(\)](#), [StreamProfileList\(\)](#), and [~StreamProfileList\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[StreamProfile.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::TemporalFilter Member List

This is the complete list of members for **ob::TemporalFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getDiffScaleRange()</b>	<b>ob::TemporalFilter</b>	inline
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>getWeightRange()</b>	<b>ob::TemporalFilter</b>	inline
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virtual
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt;const Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt;Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setDiffScale(float value)</b>	<b>ob::TemporalFilter</b>	inline
<b>setWeight(float value)</b>	<b>ob::TemporalFilter</b>	inline
<b>TemporalFilter(const std::string &amp;activationKey="")</b>	<b>ob::TemporalFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~TemporalFilter() noexcept override=default</b>	<b>ob::TemporalFilter</b>	

# ob::TemporalFilter Class Reference

Temporal filter. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::TemporalFilter:

## Public Member Functions

**TemporalFilter** (const std::string &activationKey="")

**~TemporalFilter** () noexcept override=default

**OBFloatPropertyRange getDiffScaleRange ()**

Get the **TemporalFilter** diffscale range.

void **setDiffScale** (float value)

Set the **TemporalFilter** diffscale value.

**OBFloatPropertyRange getWeightRange ()**

Get the **TemporalFilter** weight range.

void **setWeight** (float value)

Set the **TemporalFilter** weight value.

Public Member Functions inherited from **ob::Filter**

## Additional Inherited Members

Protected Member Functions inherited from **ob::Filter**

Protected Attributes inherited from **ob::Filter**

## Detailed Description

Temporal filter.

Definition at line **889** of file **Filter.hpp**.

## Constructor & Destructor Documentation

◆ **TemporalFilter()**

ob::TemporalFilter::TemporalFilter ( const std::string & activationKey = "" )

inline

Definition at line **891** of file [Filter.hpp](#).

◆ ~TemporalFilter()

ob::TemporalFilter::~TemporalFilter ( )

override default noexcept

## Member Function Documentation

◆ getDiffScaleRange()

[OBFloatPropertyRange](#) ob::TemporalFilter::getDiffScaleRange ( )

inline

Get the **TemporalFilter** diffscale range.

**Returns**

[OBFloatPropertyRange](#) the diffscale value of property range.

Definition at line **905** of file [Filter.hpp](#).

Referenced by [getDiffScaleRange\(\)](#).

◆ setDiffScale()

void ob::TemporalFilter::setDiffScale ( float value )

inline

Set the **TemporalFilter** diffscale value.

**Parameters**

**value** diffscale value.

Definition at line **922** of file [Filter.hpp](#).

◆ getWeightRange()

**OBFloatPropertyRange** ob::TemporalFilter::getWeightRange ( )

inline

Get the **TemporalFilter** weight range.

#### Returns

**OBFloatPropertyRange** the weight value of property range.

Definition at line **931** of file **Filter.hpp**.

#### ◆ setWeight()

void ob::TemporalFilter::setWeight ( float value )

inline

Set the **TemporalFilter** weight value.

#### Parameters

**value** weight value.

Definition at line **948** of file **Filter.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

## ob::ThresholdFilter Member List

This is the complete list of members for **ob::ThresholdFilter**, including all inherited members.

<b>as()</b>	<b>ob::Filter</b>	inline
<b>callback_</b>	<b>ob::Filter</b>	protected
<b>configSchemaVec_</b>	<b>ob::Filter</b>	protected
<b>enable(bool enable) const</b>	<b>ob::Filter</b>	inline virtual
<b>Filter()=default</b>	<b>ob::Filter</b>	protected
<b>Filter(ob_filter *impl)</b>	<b>ob::Filter</b>	inline explicit
<b>getConfigSchema() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigSchemaVec() const</b>	<b>ob::Filter</b>	inline virtual
<b>getConfigValue(const std::string &amp;configName) const</b>	<b>ob::Filter</b>	inline virtual
<b>getImpl() const</b>	<b>ob::Filter</b>	inline
<b>getMaxRange()</b>	<b>ob::ThresholdFilter</b>	inline
<b>getMinRange()</b>	<b>ob::ThresholdFilter</b>	inline
<b>getName() const</b>	<b>ob::Filter</b>	inline virtual
<b>impl_</b>	<b>ob::Filter</b>	protected
<b>init(ob_filter *impl)</b>	<b>ob::Filter</b>	inline protected virtual
<b>is()</b>	<b>ob::Filter</b>	
<b>isEnabled() const</b>	<b>ob::Filter</b>	inline virtual
<b>name_</b>	<b>ob::Filter</b>	protected
<b>process(std::shared_ptr&lt;const Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>pushFrame(std::shared_ptr&lt;Frame&gt; frame) const</b>	<b>ob::Filter</b>	inline virtual
<b>reset() const</b>	<b>ob::Filter</b>	inline virtual
<b>setCallBack(FilterCallback callback)</b>	<b>ob::Filter</b>	inline virtual
<b>setConfigValue(const std::string &amp;configName, double value) const</b>	<b>ob::Filter</b>	inline virtual
<b>setValueRange(uint16_t min, uint16_t max)</b>	<b>ob::ThresholdFilter</b>	inline
<b>ThresholdFilter()</b>	<b>ob::ThresholdFilter</b>	inline
<b>type()</b>	<b>ob::Filter</b>	inline virtual
<b>~Filter() noexcept</b>	<b>ob::Filter</b>	inline virtual
<b>~ThresholdFilter() noexcept override=default</b>	<b>ob::ThresholdFilter</b>	virtual

# ob::ThresholdFilter Class Reference

Creates depth Thresholding filter By controlling min and max options on the block. [More...](#)

```
#include <Filter.hpp>
```

Inheritance diagram for ob::ThresholdFilter:

## Public Member Functions

[\*\*ThresholdFilter\(\)\*\*](#)

virtual [\*\*~ThresholdFilter\(\)\*\*](#) noexcept override=default

[\*\*OBIntPropertyRange getMinRange\(\)\*\*](#)

Get the threshold filter min range.

[\*\*OBIntPropertyRange getMaxRange\(\)\*\*](#)

Get the threshold filter max range.

bool [\*\*setValueRange\(uint16\\_t min, uint16\\_t max\)\*\*](#)

Set the threshold filter max and min range.

Public Member Functions inherited from [ob::Filter](#)

## Additional Inherited Members

Protected Member Functions inherited from [ob::Filter](#)

Protected Attributes inherited from [ob::Filter](#)

## Detailed Description

Creates depth Thresholding filter By controlling min and max options on the block.

Definition at line [613](#) of file [Filter.hpp](#).

## Constructor & Destructor Documentation

◆ [\*\*ThresholdFilter\(\)\*\*](#)

ob::ThresholdFilter::ThresholdFilter( )

inline

Definition at line [615](#) of file [Filter.hpp](#).

◆ ~ThresholdFilter()

virtual ob::ThresholdFilter::~ThresholdFilter ( )

override virtual default noexcept

## Member Function Documentation

◆ getMinRange()

**OBIntPropertyRange** ob::ThresholdFilter::getMinRange ( )

inline

Get the threshold filter min range.

**Returns**

**OBIntPropertyRange** The range of the threshold filter min.

Definition at line **629** of file **Filter.hpp**.

Referenced by **getMaxRange()**.

◆ getMaxRange()

**OBIntPropertyRange** ob::ThresholdFilter::getMaxRange ( )

inline

Get the threshold filter max range.

**Returns**

**OBIntPropertyRange** The range of the threshold filter max.

Definition at line **646** of file **Filter.hpp**.

◆ setValueRange()

bool ob::ThresholdFilter::setValueRange ( uint16\_t min,

                  uint16\_t max )

inline

Set the threshold filter max and min range.

Definition at line **661** of file **Filter.hpp**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/**Filter.hpp**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::TypeHelper Member List

This is the complete list of members for **ob::TypeHelper**, including all inherited members.

<b>convertOBAccelFullScaleRangeToString</b> (const OBAccelFullScaleRange &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBFormatTypeToString</b> (const OBFormat &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBFrameMetadataTypeToString</b> (const OBFrameMetadataType &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBFrameTypeToString</b> (const OBFrameType &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBGyroFullScaleRangeToString</b> (const OBGyroFullScaleRange &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBIMUSampleRateTypeToString</b> (const OBIMUSampleRate &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBSensorTypeToString</b> (const OBSensorType &type)	<b>ob::TypeHelper</b>	inline
<b>convertOBStreamTypeToString</b> (const OBStreamType &type)	<b>ob::TypeHelper</b>	inline
<b>convertSensorTypeToStreamType</b> (OBSensorType type)	<b>ob::TypeHelper</b>	inline
<b>isVideoSensorType</b> (OBSensorType type)	<b>ob::TypeHelper</b>	inline
<b>isVideoStreamType</b> (OBStreamType type)	<b>ob::TypeHelper</b>	inline

# ob::TypeHelper Class Reference

```
#include <TypeHelper.hpp>
```

## Static Public Member Functions

```
static std::string convertOBFormatTypeToString (const OBFormat &type)
    Convert OBFormat to " string " type and then return.

static std::string convertOBFrameTypeToString (const OBFrameType &type)
    Convert OBFrameType to " string " type and then return.

static std::string convertOBStreamTypeToString (const OBStreamType &type)
    Convert OBStreamType to " string " type and then return.

static std::string convertOBSensorTypeToString (const OBSensorType &type)
    Convert OBSensorType to " string " type and then return.

static std::string convertOBIMUSampleRateTypeToString (const OBIMUSampleRate &type)
    Convert OBIMUSampleRate to " string " type and then return.

static std::string convertOBGyroFullScaleRangeTypeToString (const OBGyroFullScaleRange
&type)
    Convert OBGyroFullScaleRange to " string " type and then return.

static std::string convertOBAccelFullScaleRangeTypeToString (const OBAccelFullScaleRange
&type)
    Convert OBAccelFullScaleRange to " string " type and then return.

static std::string convertOBFrameMetadataTypeToString (const OBFrameMetadataType
&type)
    Convert OBFrameMetadataType to " string " type and then return.

static OBStreamType convertSensorTypeToStreamType (OBSensorType type)
    Convert OBSensorType to OBStreamType type and then return.

static bool isVideoSensorType (OBSensorType type)
    Check if the given sensor type is a video sensor.

static bool isVideoStreamType (OBStreamType type)
    Check if the given stream type is a video stream.
```

## Detailed Description

Definition at line 14 of file [TypeHelper.hpp](#).

## Member Function Documentation

### ◆ convertOBFormatTypeToString()

std::string ob::TypeHelper::convertOBFormatTypeToString ( const **OBFormat** & type )

inline static

Convert **OBFormat** to " string " type and then return.

#### Parameters

[in] type **OBFormat** type.

#### Returns

**OBFormat** of "string" type.

Definition at line **22** of file **TypeHelper.hpp**.

### ◆ convertOBFrameTypeToString()

std::string ob::TypeHelper::convertOBFrameTypeToString ( const **OBFrameType** & type )

inline static

Convert **OBFrameType** to " string " type and then return.

#### Parameters

[in] type **OBFrameType** type.

#### Returns

**OBFrameType** of "string" type.

Definition at line **32** of file **TypeHelper.hpp**.

### ◆ convertOBStreamTypeToString()

```
std::string ob::TypeHelper::convertOBStreamTypeToString ( const OBStreamType & type )           inline static
```

Convert **OBStreamType** to " string " type and then return.

#### Parameters

[in] type **OBStreamType** type.

#### Returns

**OBStreamType** of "string" type.

Definition at line 42 of file [TypeHelper.hpp](#).

### ◆ convertOBSensorTypeToString()

```
std::string ob::TypeHelper::convertOBSensorTypeToString ( const OBSensorType & type )           inline static
```

Convert **OBSensorType** to " string " type and then return.

#### Parameters

[in] type **OBSensorType** type.

#### Returns

**OBSensorType** of "string" type.

Definition at line 52 of file [TypeHelper.hpp](#).

### ◆ convertOBIMUSampleRateTypeToString()

```
std::string  
ob::TypeHelper::convertOBIMUSampleRateTypeToString ( const OBIMUSampleRate & type )           inline static
```

Convert **OBIMUSampleRate** to " string " type and then return.

#### Parameters

[in] type **OBIMUSampleRate** type.

#### Returns

**OBIMUSampleRate** of "string" type.

Definition at line 62 of file [TypeHelper.hpp](#).

### ◆ convertOBGyroFullScaleRangeTypeToString()

```
std::string  
ob::TypeHelper::convertOBGyroFullScaleRangeTypeToString ( const OBGyroFullScaleRange & type )    inline static
```

Convert **OBGyroFullScaleRange** to " string " type and then return.

#### Parameters

[in] type **OBGyroFullScaleRange** type.

#### Returns

**OBGyroFullScaleRange** of "string" type.

Definition at line [72](#) of file **TypeHelper.hpp**.

### ◆ convertOBAccelFullScaleRangeTypeToString()

```
std::string  
ob::TypeHelper::convertOBAccelFullScaleRangeTypeToString ( const OBAccelFullScaleRange & type )    inline static
```

Convert **OBAccelFullScaleRange** to " string " type and then return.

#### Parameters

[in] type **OBAccelFullScaleRange** type.

#### Returns

**OBAccelFullScaleRange** of "string" type.

Definition at line [82](#) of file **TypeHelper.hpp**.

### ◆ convertOBFrameMetadataTypeToString()

```
std::string  
ob::TypeHelper::convertOBFrameMetadataTypeToString ( const OBFrameMetadataType & type )    inline static
```

Convert **OBFrameMetadataType** to " string " type and then return.

#### Parameters

[in] type **OBFrameMetadataType** type.

#### Returns

**OBFrameMetadataType** of "string" type.

Definition at line [92](#) of file **TypeHelper.hpp**.

- ◆ convertSensorTypeToStreamType()

**OBStreamType** ob::TypeHelper::convertSensorTypeToStreamType ( **OBSensorType** type )

inline static

Convert **OBSensorType** to **OBStreamType** type and then return.

#### Parameters

[in] type **OBSensorType** type.

#### Returns

**OBStreamType** type.

Definition at line 102 of file **TypeHelper.hpp**.

Referenced by **ob::Config::disableStream()**, **ob::Config::enableStream()**, and  
**ob::Config::enableVideoStream()**.

- ◆ isVideoSensorType()

bool ob::TypeHelper::isVideoSensorType ( **OBSensorType** type )

inline static

Check if the given sensor type is a video sensor.

Video sensors are sensors that produce video frames.

The following sensor types are considered video sensors: OB\_SENSOR\_COLOR, OB\_SENSOR\_DEPTH,  
OB\_SENSOR\_IR, OB\_SENSOR\_IR\_LEFT, OB\_SENSOR\_IR\_RIGHT, OB\_SENSOR\_CONFIDENCE,

#### Parameters

**type** The sensor type

#### Returns

true

false

Definition at line 121 of file **TypeHelper.hpp**.

- ◆ isVideoStreamType()

```
bool ob::TypeHelper::isVideoStreamType ( OBStreamType type )
```

inline static

Check if the given stream type is a video stream.

Video streams are streams that contain video frames.

The following stream types are considered video streams: OB\_STREAM\_VIDEO, OB\_STREAM\_DEPTH, OB\_STREAM\_COLOR, OB\_STREAM\_IR, OB\_STREAM\_IR\_LEFT, OB\_STREAM\_IR\_RIGHT, OB\_STREAM\_CONFIDENCE,

## Parameters

**type** The stream type to check.

## Returns

true if the given stream type is a video stream, false otherwise.

Definition at line **140** of file [TypeHelper.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[TypeHelper.hpp](#)

## ob::Version Member List

This is the complete list of members for **ob::Version**, including all inherited members.

<b>getMajor()</b>	<b>ob::Version</b>	inline	static
<b>getMinor()</b>	<b>ob::Version</b>	inline	static
<b>getPatch()</b>	<b>ob::Version</b>	inline	static
<b>getStageVersion()</b>	<b>ob::Version</b>	inline	static
<b>getVersion()</b>	<b>ob::Version</b>	inline	static

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# ob::Version Class Reference

```
#include <Version.hpp>
```

## Static Public Member Functions

```
static int getVersion ()  
    Get the full version number of the SDK.  
  
static int getMajor ()  
    Get the major version number of the SDK.  
  
static int getMinor ()  
    Get the minor version number of the SDK.  
  
static int getPatch ()  
    Get the patch version number of the SDK.  
  
static const char * getStageVersion ()  
    Get the stage version of the SDK.
```

## Detailed Description

Definition at line **13** of file **Version.hpp**.

## Member Function Documentation

### ◆ **getVersion()**

int ob::Version::getVersion ( )

inline static

Get the full version number of the SDK.

The full version number equals to: major \* 10000 + minor \* 100 + patch

#### Returns

int The full version number of the SDK.

Definition at line **21** of file **Version.hpp**.

### ◆ **getMajor()**

```
int ob::Version::getMajor( )
```

inline static

Get the major version number of the SDK.

#### Returns

int The major version number of the SDK.

Definition at line [29](#) of file [Version.hpp](#).

#### ◆ [getMinor\(\)](#)

```
int ob::Version::getMinor( )
```

inline static

Get the minor version number of the SDK.

#### Returns

int The minor version number of the SDK.

Definition at line [38](#) of file [Version.hpp](#).

#### ◆ [getPatch\(\)](#)

```
int ob::Version::getPatch( )
```

inline static

Get the patch version number of the SDK.

#### Returns

int The patch version number of the SDK.

Definition at line [47](#) of file [Version.hpp](#).

#### ◆ [getStageVersion\(\)](#)

```
const char * ob::Version::getStageVersion ( )
```

inline static

Get the stage version of the SDK.

The stage version string is a vendor defined string for some special releases.

### Returns

char\* The stage version string of the SDK.

Definition at line [57](#) of file [Version.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Version.hpp](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::VideoFrame Member List

This is the complete list of members for **ob::VideoFrame**, including all inherited members.

<b>as()</b>	<b>ob::Frame</b>	inline
<b>as() const</b>	<b>ob::Frame</b>	inline
<b>data() const</b>	<b>ob::Frame</b>	inline virtual
<b>dataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>format() const</b>	<b>ob::Frame</b>	inline virtual
<b>Frame(const ob_frame *impl)</b>	<b>ob::Frame</b>	inline explicit
<b>getData() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDataSize() const</b>	<b>ob::Frame</b>	inline virtual
<b>getDevice() const</b>	<b>ob::Frame</b>	inline
<b>getFormat() const</b>	<b>ob::Frame</b>	inline virtual
<b>getGlobalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getHeight() const</b>	<b>ob::VideoFrame</b>	inline
<b>getImpl() const</b>	<b>ob::Frame</b>	inline
<b>getIndex() const</b>	<b>ob::Frame</b>	inline virtual
<b>getMetadata() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataSize() const</b>	<b>ob::Frame</b>	inline
<b>getMetadataValue(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>getPixelAvailableBitSize() const</b>	<b>ob::VideoFrame</b>	inline
<b>getPixelType() const</b>	<b>ob::VideoFrame</b>	inline
<b>getSensor() const</b>	<b>ob::Frame</b>	inline
<b>getStreamProfile() const</b>	<b>ob::Frame</b>	inline
<b>getSystemTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>getType() const</b>	<b>ob::Frame</b>	inline virtual
<b>getWidth() const</b>	<b>ob::VideoFrame</b>	inline
<b>globalTimeStampUs() const</b>	<b>ob::Frame</b>	inline
<b>hasMetadata(OBFrameMetadataType type) const</b>	<b>ob::Frame</b>	inline
<b>height() const</b>	<b>ob::VideoFrame</b>	inline
<b>impl_</b>	<b>ob::Frame</b>	protected
<b>index() const</b>	<b>ob::Frame</b>	inline virtual
<b>is() const</b>	<b>ob::Frame</b>	
<b>metadata() const</b>	<b>ob::Frame</b>	inline

<b>metadataSize()</b> const	<b>ob::Frame</b>	inline
<b>pixelAvailableBitSize()</b> const	<b>ob::VideoFrame</b>	inline
<b>systemTimeStamp()</b> const	<b>ob::Frame</b>	inline
<b>systemTimeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>timeStamp()</b> const	<b>ob::Frame</b>	inline
<b>timeStampUs()</b> const	<b>ob::Frame</b>	inline
<b>type()</b> const	<b>ob::Frame</b>	inline
<b>VideoFrame</b> (const ob_frame *impl)	<b>ob::VideoFrame</b>	inline explicit
<b>width()</b> const	<b>ob::VideoFrame</b>	inline
<b>~Frame()</b> noexcept	<b>ob::Frame</b>	inline virtual
<b>~VideoFrame()</b> noexcept override=default	<b>ob::VideoFrame</b>	

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob::VideoFrame Class Reference

Define the **VideoFrame** class, which inherits from the **Frame** class. [More...](#)

```
#include <Frame.hpp>
```

Inheritance diagram for ob::VideoFrame:

## Public Member Functions

**VideoFrame** (const **ob\_frame** \*impl)

Construct a new **VideoFrame** object with a given pointer to the internal frame object.

**~VideoFrame** () noexcept override=default

**uint32\_t getWidth** () const

Get the width of the frame.

**uint32\_t getHeight** () const

Get the height of the frame.

**OBPixelType getPixelType** () const

Get the Pixel Type object.

**uint8\_t getPixelAvailableBitSize** () const

Get the effective number of pixels in the frame.

**uint32\_t width** () const

**uint32\_t height** () const

**uint8\_t pixelAvailableBitSize** () const

Public Member Functions inherited from **ob::Frame**

## Additional Inherited Members

Protected Attributes inherited from **ob::Frame**

## Detailed Description

Define the **VideoFrame** class, which inherits from the **Frame** class.

Definition at line **392** of file **Frame.hpp**.

## Constructor & Destructor Documentation

## ◆ VideoFrame()

ob::VideoFrame::VideoFrame ( const **ob\_frame** \* impl )

inline explicit

Construct a new **VideoFrame** object with a given pointer to the internal frame object.

### Attention

After calling this constructor, the frame object will own the internal frame object, and the internal frame object will be deleted when the frame object is destroyed.

The internal frame object should not be deleted by the caller.

### Parameters

**impl** The pointer to the internal frame object.

Definition at line **403** of file **Frame.hpp**.

Referenced by **ob::ColorFrame::ColorFrame()**, **ob::ConfidenceFrame::ConfidenceFrame()**,  
**ob::DepthFrame::DepthFrame()**, and **ob::IRFrame::IRFrame()**.

## ◆ ~VideoFrame()

ob::VideoFrame::~VideoFrame ( )

override default noexcept

## Member Function Documentation

### ◆ getWidth()

uint32\_t ob::VideoFrame::getWidth ( ) const

inline

Get the width of the frame.

### Returns

uint32\_t The width of the frame.

Definition at line **412** of file **Frame.hpp**.

Referenced by **getWidth()**, and **width()**.

### ◆ getHeight()

```
uint32_t ob::VideoFrame::getHeight ( ) const
```

inline

Get the height of the frame.

#### Returns

uint32\_t The height of the frame.

Definition at line [425](#) of file [Frame.hpp](#).

Referenced by [height\(\)](#).

### ◆ [getPixelType\(\)](#)

```
OBPixelType ob::VideoFrame::getPixelType ( ) const
```

inline

Get the Pixel Type object.

Usually used to determine the pixel type of depth frame (depth, disparity, raw phase, etc.)

#### Attention

Always return OB\_PIXEL\_UNKNOWN for non-depth frame currently

#### Returns

[OBPixelType](#)

Definition at line [441](#) of file [Frame.hpp](#).

### ◆ [getPixelAvailableBitSize\(\)](#)

```
uint8_t ob::VideoFrame::getPixelAvailableBitSize( ) const
```

inline

Get the effective number of pixels in the frame.

### Attention

Only valid for Y8/Y10/Y11/Y12/Y14/Y16 format.

### Returns

uint8\_t The effective number of pixels in the frame, or 0 if it is an unsupported format.

Definition at line [455](#) of file [Frame.hpp](#).

Referenced by [pixelAvailableBitSize\(\)](#).

### ◆ width()

```
uint32_t ob::VideoFrame::width( ) const
```

inline

Definition at line [465](#) of file [Frame.hpp](#).

Referenced by [getWidth\(\)](#).

### ◆ height()

```
uint32_t ob::VideoFrame::height( ) const
```

inline

Definition at line [469](#) of file [Frame.hpp](#).

Referenced by [getHeight\(\)](#).

### ◆ pixelAvailableBitSize()

```
uint8_t ob::VideoFrame::pixelAvailableBitSize( ) const
```

inline

Definition at line [473](#) of file [Frame.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Frame.hpp](#)



## ob::VideoStreamProfile Member List

This is the complete list of members for **ob::VideoStreamProfile**, including all inherited members.

<b>as()</b>	<b>ob::StreamProfile</b>
<b>as() const</b>	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo(std::shared_ptr&lt; StreamProfile &gt; target, const OBExtrinsic &amp;extrinsic)</b>	<b>ob::StreamProfile</b>
<b>bindExtrinsicTo(const OBStreamType &amp;targetStreamType, const OBExtrinsic &amp;extrinsic)</b>	<b>ob::StreamProfile</b>
<b>format() const</b>	<b>ob::StreamProfile</b>
<b>fps() const</b>	<b>ob::VideoStreamF</b>
<b>getDistortion() const</b>	<b>ob::VideoStreamF</b>
<b>getExtrinsicTo(std::shared_ptr&lt; StreamProfile &gt; target) const</b>	<b>ob::StreamProfile</b>
<b>getFormat() const</b>	<b>ob::StreamProfile</b>
<b>getFps() const</b>	<b>ob::VideoStreamF</b>
<b>getHeight() const</b>	<b>ob::VideoStreamF</b>
<b>getImpl() const</b>	<b>ob::StreamProfile</b>
<b>getIntrinsic() const</b>	<b>ob::VideoStreamF</b>
<b>getType() const</b>	<b>ob::StreamProfile</b>
<b>getWidth() const</b>	<b>ob::VideoStreamF</b>
<b>height() const</b>	<b>ob::VideoStreamF</b>
<b>impl_</b>	<b>ob::StreamProfile</b>
<b>is() const</b>	<b>ob::StreamProfile</b>
<b>operator=(StreamProfile &amp;streamProfile)=delete</b>	<b>ob::StreamProfile</b>
<b>operator=(StreamProfile &amp;&amp;streamProfile) noexcept</b>	<b>ob::StreamProfile</b>
<b>setDistortion(const OBCameraDistortion &amp;distortion)</b>	<b>ob::VideoStreamF</b>
<b>setIntrinsic(const OBCameraIntrinsic &amp;intrinsic)</b>	<b>ob::VideoStreamF</b>
<b>StreamProfile(StreamProfile &amp;streamProfile)=delete</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(StreamProfile &amp;&amp;streamProfile) noexcept</b>	<b>ob::StreamProfile</b>
<b>StreamProfile(const ob_stream_profile_t *impl)</b>	<b>ob::StreamProfile</b>
<b>type() const</b>	<b>ob::StreamProfile</b>
<b>VideoStreamProfile(const ob_stream_profile_t *impl)</b>	<b>ob::VideoStreamF</b>
<b>width() const</b>	<b>ob::VideoStreamF</b>
<b>~StreamProfile() noexcept</b>	<b>ob::StreamProfile</b>
<b>~VideoStreamProfile() noexcept override=default</b>	<b>ob::VideoStreamF</b>



# ob::VideoStreamProfile Class Reference

Class representing a video stream profile. [More...](#)

```
#include <StreamProfile.hpp>
```

Inheritance diagram for ob::VideoStreamProfile:

## Public Member Functions

**VideoStreamProfile** (const ob\_stream\_profile\_t \*impl)

**~VideoStreamProfile** () noexcept override=default

uint32\_t **getFps** () const

Return the frame rate of the stream.

uint32\_t **getWidth** () const

Return the width of the stream.

uint32\_t **getHeight** () const

Return the height of the stream.

**OBCameraIntrinsic** **getIntrinsic** () const

Get the intrinsic parameters of the stream.

void **setIntrinsic** (const **OBCameraIntrinsic** &intrinsic)

Set the intrinsic parameters of the stream.

**OBCameraDistortion** **getDistortion** () const

Get the distortion parameters of the stream.

void **setDistortion** (const **OBCameraDistortion** &distortion)

Set the distortion parameters of the stream.

uint32\_t **fps** () const

uint32\_t **width** () const

uint32\_t **height** () const

Public Member Functions inherited from **ob::StreamProfile**

## Additional Inherited Members

Protected Member Functions inherited from **ob::StreamProfile**

Protected Attributes inherited from **ob::StreamProfile**

## Detailed Description

Class representing a video stream profile.

Definition at line **165** of file [StreamProfile.hpp](#).

## Constructor & Destructor Documentation

### ◆ VideoStreamProfile()

ob::VideoStreamProfile::VideoStreamProfile ( const ob\_stream\_profile\_t \* impl )

inline explicit

Definition at line **167** of file [StreamProfile.hpp](#).

### ◆ ~VideoStreamProfile()

ob::VideoStreamProfile::~VideoStreamProfile ( )

override default noexcept

## Member Function Documentation

### ◆ getFps()

uint32\_t ob::VideoStreamProfile::getFps ( ) const

inline

Return the frame rate of the stream.

#### Returns

uint32\_t Return the frame rate of the stream.

Definition at line **176** of file [StreamProfile.hpp](#).

Referenced by [fps\(\)](#), and [getFps\(\)](#).

### ◆ getWidth()

uint32\_t ob::VideoStreamProfile::getWidth ( ) const inline

Return the width of the stream.

#### Returns

uint32\_t Return the width of the stream.

Definition at line [188](#) of file [StreamProfile.hpp](#).

Referenced by [width\(\)](#).

#### ◆ [getHeight\(\)](#)

uint32\_t ob::VideoStreamProfile::getHeight ( ) const inline

Return the height of the stream.

#### Returns

uint32\_t Return the height of the stream.

Definition at line [200](#) of file [StreamProfile.hpp](#).

Referenced by [height\(\)](#).

#### ◆ [getIntrinsic\(\)](#)

**OBCameraIntrinsic** ob::VideoStreamProfile::getIntrinsic ( ) const inline

Get the intrinsic parameters of the stream.

#### Returns

**OBCameraIntrinsic** Return the intrinsic parameters.

Definition at line [212](#) of file [StreamProfile.hpp](#).

#### ◆ [setIntrinsic\(\)](#)

```
void ob::VideoStreamProfile::setIntrinsic ( const OBCameraIntrinsic & intrinsic )
```

inline

Set the intrinsic parameters of the stream.

### Parameters

**intrinsic** The intrinsic parameters.

Definition at line **224** of file **StreamProfile.hpp**.

### ◆ **getDistortion()**

```
OBCameraDistortion ob::VideoStreamProfile::getDistortion ( ) const
```

inline

Get the distortion parameters of the stream.

Brown distortion model

### Returns

**OBCameraDistortion** Return the distortion parameters.

Definition at line **236** of file **StreamProfile.hpp**.

### ◆ **setDistortion()**

```
void ob::VideoStreamProfile::setDistortion ( const OBCameraDistortion & distortion )
```

inline

Set the distortion parameters of the stream.

### Parameters

**distortion** The distortion parameters.

Definition at line **248** of file **StreamProfile.hpp**.

### ◆ **fps()**

```
uint32_t ob::VideoStreamProfile::fps ( ) const
```

inline

Definition at line **256** of file **StreamProfile.hpp**.

Referenced by **getFps()**.

◆ **width()**

uint32\_t ob::VideoStreamProfile::width ( ) const

inline

Definition at line **260** of file [StreamProfile.hpp](#).

Referenced by [getWidth\(\)](#).

◆ **height()**

uint32\_t ob::VideoStreamProfile::height ( ) const

inline

Definition at line **264** of file [StreamProfile.hpp](#).

Referenced by [getHeight\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[StreamProfile.hpp](#)

# Deprecated List

## Member **ob::Config::enableAllStream ()**

Use enableStream(std::shared\_ptr<StreamProfile> streamProfile) instead

## Struct **OBDeviceSyncConfig**

This structure is deprecated, please use **ob\_multi\_device\_sync\_config** instead

## Member **OBSyncMode**

This define is deprecated, please use **ob\_multi\_device\_sync\_mode** instead

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# hpp Directory Reference

## Files

### [Context.hpp](#)

The SDK context class, which serves as the entry point to the underlying SDK. It is used to query device lists, handle device callbacks, and set the log level.

### [Device.hpp](#)

Device related types, including operations such as getting and creating a device, setting and obtaining device attributes, and obtaining sensors.

### [Error.hpp](#)

This file defines the Error class, which describes abnormal errors within the SDK. Detailed information about the exception can be obtained through this class.

### [Filter.hpp](#)

This file contains the Filter class, which is the processing unit of the SDK that can perform point cloud generation, format conversion, and other functions.

### [Frame.hpp](#)

Frame related type, which is mainly used to obtain frame data and frame information.

### [Pipeline.hpp](#)

The SDK's advanced API type can quickly implement switching streaming and frame synchronization operations.

### [RecordPlayback.hpp](#)

Record and playback device-related types, including interfaces to create recording and playback devices, record and playback streaming data, etc.

### [Sensor.hpp](#)

Defines types related to sensors, which are used to obtain stream configurations, open and close streams, and set and get sensor properties.

### [StreamProfile.hpp](#)

The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream.

### [TypeHelper.hpp](#)

### [Types.hpp](#)

### [Utils.hpp](#)

The SDK utils class.

### [Version.hpp](#)

Provides functions to retrieve version information of the SDK.



# libobsensor Directory Reference

## Directories

[\*\*h\*\*](#)

[\*\*hpp\*\*](#)

## Files

### [\*\*ObSensor.h\*\*](#)

This file serves as the C entrance for the OrbbecSDK library. It includes all necessary header files for OrbbecSDK usage.

### [\*\*ObSensor.hpp\*\*](#)

This is the main entry point for the OrbbecSDK C++ library. It includes all necessary header files for using the library.

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# h Directory Reference

## Files

### **Advanced.h**

### **Context.h**

Context is a management class that describes the runtime of the SDK and is responsible for resource allocation and release of the SDK. Context has the ability to manage multiple devices. It is responsible for enumerating devices, monitoring device callbacks, and enabling multi-device synchronization.

### **Device.h**

Device-related functions, including operations such as obtaining and creating a device, setting and obtaining device property, and obtaining sensors.

### **Error.h**

Functions for handling errors, mainly used for obtaining error messages.

### **Export.h**

### **Filter.h**

The processing unit of the SDK can perform point cloud generation, format conversion and other functions.

### **Frame.h**

Frame related function is mainly used to obtain frame data and frame information.

### **MultipleDevices.h**

This file contains the multiple devices related API which is used to control the synchronization between multiple devices and the synchronization between different sensor within single device.

### **ObTypes.h**

Provide structs commonly used in the SDK, enumerating constant definitions.

### **Pipeline.h**

The SDK's advanced API can quickly implement functions such as switching streaming, frame synchronization, software filtering, etc., suitable for applications, and the algorithm focuses on rgbd data stream scenarios. If you are on real-time or need to handle synchronization separately, align the scene. Please use the interface of Device's Lower API.

### **Property.h**

Control command property list maintenance.

### **RecordPlayback.h**

### **Sensor.h**

Defines types related to sensors, used for obtaining stream configurations, opening and closing streams, and setting and getting sensor properties.

### **StreamProfile.h**

The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream.

**TypeHelper.h**

**Utils.h**

**Version.h**

Functions for retrieving the SDK version number information.

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# include Directory Reference

## Directories

[\*\*libobsensor\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# File List

Here is a list of all files with brief descriptions:

[detail level 1 2 3 4]

## include

### [libobsensor](#)

#### [h](#)

##### [Advanced.h](#)

##### [Context.h](#)

##### [Device.h](#)

##### [Error.h](#)

##### [Export.h](#)

##### [Filter.h](#)

##### [Frame.h](#)

##### [MultipleDevices.h](#)

##### [ObTypes.h](#)

##### [Pipeline.h](#)

##### [Property.h](#)

##### [RecordPlayback.h](#)

##### [Sensor.h](#)

Context is a management class that describes the runtime of the SDK and is responsible for resource allocation and release of the SDK. Context has the ability to manage multiple devices. It is responsible for enumerating devices, monitoring device callbacks, and enabling multi-device synchronization

Device-related functions, including operations such as obtaining and creating a device, setting and obtaining device property, and obtaining sensors

Functions for handling errors, mainly used for obtaining error messages

The processing unit of the SDK can perform point cloud generation, format conversion and other functions

Frame related function is mainly used to obtain frame data and frame information

This file contains the multiple devices related API which is used to control the synchronization between multiple devices and the synchronization between different sensor within single device

Provide structs commonly used in the SDK, enumerating constant definitions

The SDK's advanced API can quickly implement functions such as switching streaming, frame synchronization, software filtering, etc., suitable for applications, and the algorithm focuses on rgbd data stream scenarios. If you are on real-time or need to handle synchronization separately, align the scene. Please use the interface of Device's Lower API

Control command property list maintenance

Defines types related to sensors, used for obtaining stream configurations, opening and closing streams, and setting and getting sensor properties

<b>StreamProfile.h</b>	The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream
<b>TypeHelper.h</b>	
<b>Utils.h</b>	
<b>Version.h</b>	Functions for retrieving the SDK version number information
<b>hpp</b>	
<b>Context.hpp</b>	The SDK context class, which serves as the entry point to the underlying SDK. It is used to query device lists, handle device callbacks, and set the log level
<b>Device.hpp</b>	Device related types, including operations such as getting and creating a device, setting and obtaining device attributes, and obtaining sensors
<b>Error.hpp</b>	This file defines the Error class, which describes abnormal errors within the SDK. Detailed information about the exception can be obtained through this class
<b>Filter.hpp</b>	This file contains the Filter class, which is the processing unit of the SDK that can perform point cloud generation, format conversion, and other functions
<b>Frame.hpp</b>	Frame related type, which is mainly used to obtain frame data and frame information
<b>Pipeline.hpp</b>	The SDK's advanced API type can quickly implement switching streaming and frame synchronization operations
<b>RecordPlayback.hpp</b>	Record and playback device-related types, including interfaces to create recording and playback devices, record and playback streaming data, etc
<b>Sensor.hpp</b>	Defines types related to sensors, which are used to obtain stream configurations, open and close streams, and set and get sensor properties
<b>StreamProfile.hpp</b>	The stream profile related type is used to get information such as the width, height, frame rate, and format of the stream
<b>TypeHelper.hpp</b>	
<b>Types.hpp</b>	
<b>Utils.hpp</b>	The SDK utils class
<b>Version.hpp</b>	Provides functions to retrieve version information of the SDK
<b>ObSensor.h</b>	This file serves as the C entrance for the OrbbecSDK library. It includes all necessary header files for OrbbecSDK usage
<b>ObSensor.hpp</b>	This is the main entry point for the OrbbecSDK C++ library. It includes all necessary header files for using the library

Here is a list of all class members with links to the classes they belong to:

- a -

- AccelFrame() : [ob::AccelFrame](#)
- AccelStreamProfile() : [ob::AccelStreamProfile](#)
- address : [OBNetIpConfig](#)
- Align() : [ob::Align](#)
- alpha : [OBSpatialAdvancedFilterParams](#)
- args : [ob\\_error](#)
- as() : [ob::Filter](#), [ob::Frame](#), [ob::StreamProfile](#)
- asicName() : [ob::DeviceInfo](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- b -

- b : [OBColorPoint](#)
- baseline : [BASELINE\\_CALIBRATION\\_PARAM](#), [OBDisparityParam](#)
- bias : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- bindExtrinsicTo() : [ob::StreamProfile](#)
- bitSize : [OBDisparityParam](#)
- BufferDestroyCallback : [ob::FrameFactory](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- c -

- calibration2dTo2d() : [ob::CoordinateTransformHelper](#)
- calibration2dTo3d() : [ob::CoordinateTransformHelper](#)
- calibration3dTo2d() : [ob::CoordinateTransformHelper](#)
- calibration3dTo3d() : [ob::CoordinateTransformHelper](#)
- callback\_ : [ob::Filter](#), [ob::Sensor](#)
- CameraParamList() : [ob::CameraParamList](#)
- checksum : [OBDepthWorkMode](#)
- chipBottomTemp : [OBDeviceTemperature](#)
- chipTopTemp : [OBDeviceTemperature](#)
- colorDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- ColorFrame() : [ob::ColorFrame](#)
- colorFrame() : [ob::FrameSet](#)
- ConfidenceFrame() : [ob::ConfidenceFrame](#)
- Config() : [ob::Config](#)
- configSchemaVec\_ : [ob::Filter](#)
- connectionType() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- Context() : [ob::Context](#)
- convertOBAccelFullScaleRangeTypeToString() : [ob::TypeHelper](#)
- convertOBFormatTypeToString() : [ob::TypeHelper](#)
- convertOBFrameMetadataTypeToString() : [ob::TypeHelper](#)
- convertOBFrameTypeToString() : [ob::TypeHelper](#)
- convertOBGyroFullScaleRangeTypeToString() : [ob::TypeHelper](#)
- convertOBIMUSampleRateTypeToString() : [ob::TypeHelper](#)
- convertOBSSensorTypeToString() : [ob::TypeHelper](#)
- convertOBStreamTypeToString() : [ob::TypeHelper](#)
- convertSensorTypeToStreamType() : [ob::TypeHelper](#)
- count() : [ob::CameraParamList](#), [ob::DevicePresetList](#), [ob::OBDepthWorkModeList](#),  
[ob::OBFilterList](#), [ob::SensorList](#), [ob::StreamProfileList](#)
- cpuTemp : [OBDeviceTemperature](#)
- create() : [ob::StreamProfileFactory](#)
- createFilter() : [ob::FilterFactory](#)
- createFrame() : [ob::FrameFactory](#)
- createFrameFromBuffer() : [ob::FrameFactory](#)
- createFrameFromOtherFrame() : [ob::FrameFactory](#)
- createFrameFromStreamProfile() : [ob::FrameFactory](#)
- createNetDevice() : [ob::Context](#)
- createPrivateFilter() : [ob::FilterFactory](#)
- createRecommendedFilters() : [ob::Sensor](#)
- createVideoFrame() : [ob::FrameFactory](#)
- createVideoFrameFromBuffer() : [ob::FrameFactory](#)
- cur : [OBBoolPropertyRange](#), [OBFloatPropertyRange](#), [OBIntPropertyRange](#),

**OBUInt16PropertyRange**, **OBUInt8PropertyRange**

- cx : **OBCameraIntrinsic**
- cy : **OBCameraIntrinsic**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- d -

- data() : [ob::Frame](#), [OBDataChunk](#)
- dataSize() : [ob::Frame](#)
- DecimationFilter() : [ob::DecimationFilter](#)
- def : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#),  
[OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- depthDecimationFactor : [OBPresetResolutionConfig](#)
- depthDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- depthDistortion : [OBCameraParam](#)
- DepthFrame() : [ob::DepthFrame](#)
- depthFrame() : [ob::FrameSet](#)
- depthIntrinsic : [OBCameraParam](#)
- desc : [OBFilterConfigSchemaItem](#)
- Device() : [ob::Device](#)
- DeviceChangedCallback : [ob::Context](#)
- deviceCount() : [ob::DeviceList](#)
- DeviceFrameInterleaveList() : [ob::DeviceFrameInterleaveList](#)
- DeviceFwUpdateCallback : [ob::Device](#)
- deviceId : [OBDeviceSyncConfig](#)
- DeviceInfo() : [ob::DeviceInfo](#)
- DeviceList() : [ob::DeviceList](#)
- DevicePresetList() : [ob::DevicePresetList](#)
- deviceStateChangeCallback\_ : [ob::Device](#)
- DeviceStateChangedCallback : [ob::Device](#)
- deviceStateChangedCallback() : [ob::Device](#)
- deviceTriggerSignalOutDelay : [OBDeviceSyncConfig](#)
- deviceTriggerSignalOutPolarity : [OBDeviceSyncConfig](#)
- deviceType() : [ob::DeviceInfo](#)
- deviceUpgrade() : [ob::Device](#)
- deviceUpgradeFromData() : [ob::Device](#)
- dhcp : [OBNetIpConfig](#)
- disableAllStream() : [ob::Config](#)
- disableFrameSync() : [ob::Pipeline](#)
- disableStream() : [ob::Config](#)
- disp\_diff : [OBNoiseRemovalFilterParams](#), [OBSpatialAdvancedFilterParams](#)
- DisparityTransform() : [ob::DisparityTransform](#)
- dispIntPlace : [OBDisparityParam](#)
- dispOffset : [OBDisparityParam](#)
- distortion : [OBCalibrationParam](#)

Here is a list of all class members with links to the classes they belong to:

- e -

- enable : [HDR\\_CONFIG](#), [ob::Filter](#), [ob\\_device\\_timestamp\\_reset\\_config](#), [OBDispOffsetConfig](#)
- enable\_direction : [ob\\_margin\\_filter\\_config](#)
- enableAccelStream() : [ob::Config](#)
- enableAllStream() : [ob::Config](#)
- enableDeviceClockSync() : [ob::Context](#)
- enableFrameSync() : [ob::Pipeline](#)
- enableGlobalTimestamp() : [ob::Device](#)
- enableGyroStream() : [ob::Config](#)
- enableHeartbeat() : [ob::Device](#)
- enableNetDeviceEnumeration() : [ob::Context](#)
- enableStream() : [ob::Config](#)
- enableVideoStream() : [ob::Config](#)
- exception\_type : [ob\\_error](#)
- exportSettingsAsPresetJsonData() : [ob::Device](#)
- exportSettingsAsPresetJsonFile() : [ob::Device](#)
- exposure\_1 : [HDR\\_CONFIG](#)
- exposure\_2 : [HDR\\_CONFIG](#)
- extrinsics : [OBCalibrationParam](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- f -

- Filter() : [\*\*ob::Filter\*\*](#)
- firmwareVersion() : [\*\*ob::DeviceInfo\*\*](#)
- format() : [\*\*ob::Frame\*\*](#), [\*\*ob::StreamProfile\*\*](#)
- FormatConvertFilter() : [\*\*ob::FormatConvertFilter\*\*](#)
- fps() : [\*\*ob::VideoStreamProfile\*\*](#)
- Frame() : [\*\*ob::Frame\*\*](#)
- FrameCallback : [\*\*ob::Sensor\*\*](#)
- frameCount() : [\*\*ob::FrameSet\*\*](#)
- FrameSet() : [\*\*ob::FrameSet\*\*](#)
- FrameSetCallback : [\*\*ob::Pipeline\*\*](#)
- frameSetCallback() : [\*\*ob::Pipeline\*\*](#)
- framesPerTrigger : [\*\*ob\\_multi\\_device\\_sync\\_config\*\*](#)
- freeIdleMemory() : [\*\*ob::Context\*\*](#)
- fullDataSize : [\*\*OBDataChunk\*\*](#)
- fullScaleRange() : [\*\*ob::AccelStreamProfile\*\*](#), [\*\*ob::GyroStreamProfile\*\*](#)
- function : [\*\*ob\\_error\*\*](#)
- fwUpdateCallback\_ : [\*\*ob::Device\*\*](#)
- fx : [\*\*OBCameraIntrinsic\*\*](#), [\*\*OBDisparityParam\*\*](#)
- fy : [\*\*OBCameraIntrinsic\*\*](#)

Here is a list of all functions with links to the classes they belong to:

- a -

- AccelFrame() : [ob::AccelFrame](#)
- AccelStreamProfile() : [ob::AccelStreamProfile](#)
- Align() : [ob::Align](#)
- as() : [ob::Filter](#), [ob::Frame](#), [ob::StreamProfile](#)
- asicName() : [ob::DeviceInfo](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- b -

- bindExtrinsicTo() : [\*\*ob::StreamProfile\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- c -

- calibration2dTo2d() : [ob::CoordinateTransformHelper](#)
- calibration2dTo3d() : [ob::CoordinateTransformHelper](#)
- calibration3dTo2d() : [ob::CoordinateTransformHelper](#)
- calibration3dTo3d() : [ob::CoordinateTransformHelper](#)
- CameraParamList() : [ob::CameraParamList](#)
- ColorFrame() : [ob::ColorFrame](#)
- colorFrame() : [ob::FrameSet](#)
- ConfidenceFrame() : [ob::ConfidenceFrame](#)
- Config() : [ob::Config](#)
- connectionType() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- Context() : [ob::Context](#)
- convertOBAccelFullScaleRangeTypeToString() : [ob::TypeHelper](#)
- convertOBFormatTypeToString() : [ob::TypeHelper](#)
- convertOBFrameMetadataTypeToString() : [ob::TypeHelper](#)
- convertOBFrameTypeToString() : [ob::TypeHelper](#)
- convertOBGyroFullScaleRangeTypeToString() : [ob::TypeHelper](#)
- convertOBIMUSampleRateTypeToString() : [ob::TypeHelper](#)
- convertOBSensorTypeToString() : [ob::TypeHelper](#)
- convertOBStreamTypeToString() : [ob::TypeHelper](#)
- convertSensorTypeToStreamType() : [ob::TypeHelper](#)
- count() : [ob::CameraParamList](#), [ob::DevicePresetList](#), [ob::OBDepthWorkModeList](#),  
[ob::OBFilterList](#), [ob::SensorList](#), [ob::StreamProfileList](#)
- create() : [ob::StreamProfileFactory](#)
- createFilter() : [ob::FilterFactory](#)
- createFrame() : [ob::FrameFactory](#)
- createFrameFromBuffer() : [ob::FrameFactory](#)
- createFrameFromOtherFrame() : [ob::FrameFactory](#)
- createFrameFromStreamProfile() : [ob::FrameFactory](#)
- createNetDevice() : [ob::Context](#)
- createPrivateFilter() : [ob::FilterFactory](#)
- createRecommendedFilters() : [ob::Sensor](#)
- createVideoFrame() : [ob::FrameFactory](#)
- createVideoFrameFromBuffer() : [ob::FrameFactory](#)

Here is a list of all functions with links to the classes they belong to:

- d -

- data() : [\*\*ob::Frame\*\*](#)
- dataSize() : [\*\*ob::Frame\*\*](#)
- DecimationFilter() : [\*\*ob::DecimationFilter\*\*](#)
- DepthFrame() : [\*\*ob::DepthFrame\*\*](#)
- depthFrame() : [\*\*ob::FrameSet\*\*](#)
- Device() : [\*\*ob::Device\*\*](#)
- deviceCount() : [\*\*ob::DeviceList\*\*](#)
- DeviceFrameInterleaveList() : [\*\*ob::DeviceFrameInterleaveList\*\*](#)
- DeviceInfo() : [\*\*ob::DeviceInfo\*\*](#)
- DeviceList() : [\*\*ob::DeviceList\*\*](#)
- DevicePresetList() : [\*\*ob::DevicePresetList\*\*](#)
- deviceStateChangedCallback() : [\*\*ob::Device\*\*](#)
- deviceType() : [\*\*ob::DeviceInfo\*\*](#)
- deviceUpgrade() : [\*\*ob::Device\*\*](#)
- deviceUpgradeFromData() : [\*\*ob::Device\*\*](#)
- disableAllStream() : [\*\*ob::Config\*\*](#)
- disableFrameSync() : [\*\*ob::Pipeline\*\*](#)
- disableStream() : [\*\*ob::Config\*\*](#)
- DisparityTransform() : [\*\*ob::DisparityTransform\*\*](#)

Here is a list of all functions with links to the classes they belong to:

- e -

- enable() : [\*\*ob::Filter\*\*](#)
- enableAccelStream() : [\*\*ob::Config\*\*](#)
- enableAllStream() : [\*\*ob::Config\*\*](#)
- enableDeviceClockSync() : [\*\*ob::Context\*\*](#)
- enableFrameSync() : [\*\*ob::Pipeline\*\*](#)
- enableGlobalTimestamp() : [\*\*ob::Device\*\*](#)
- enableGyroStream() : [\*\*ob::Config\*\*](#)
- enableHeartbeat() : [\*\*ob::Device\*\*](#)
- enableNetDeviceEnumeration() : [\*\*ob::Context\*\*](#)
- enableStream() : [\*\*ob::Config\*\*](#)
- enableVideoStream() : [\*\*ob::Config\*\*](#)
- exportSettingsAsPresetJsonData() : [\*\*ob::Device\*\*](#)
- exportSettingsAsPresetJsonFile() : [\*\*ob::Device\*\*](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- f -

- Filter() : [ob::Filter](#)
- firmwareVersion() : [ob::DeviceInfo](#)
- format() : [ob::Frame](#), [ob::StreamProfile](#)
- FormatConvertFilter() : [ob::FormatConvertFilter](#)
- fps() : [ob::VideoStreamProfile](#)
- Frame() : [ob::Frame](#)
- frameCount() : [ob::FrameSet](#)
- FrameSet() : [ob::FrameSet](#)
- frameSetCallback() : [ob::Pipeline](#)
- freeIdleMemory() : [ob::Context](#)
- fullScaleRange() : [ob::AccelStreamProfile](#), [ob::GyroStreamProfile](#)

Here is a list of all functions with links to the classes they belong to:

- g -

- getAccelStreamProfile() : [ob::StreamProfileList](#)
- getAlignToStreamType() : [ob::Align](#)
- getAlphaRange() : [ob::SpatialAdvancedFilter](#)
- getArgs() : [ob::Error](#)
- getAsicName() : [ob::DeviceInfo](#)
- getAvailableFrameInterleaveList() : [ob::Device](#)
- getAvailablePresetList() : [ob::Device](#)
- getAvailablePresetResolutionConfigList() : [ob::Device](#)
- getBoolProperty() : [ob::Device](#)
- getBoolPropertyRange() : [ob::Device](#)
- getCalibrationCameraParamList() : [ob::Device](#)
- getCalibrationParam() : [ob::Pipeline](#)
- getCameraParam() : [ob::CameraParamList](#), [ob::Pipeline](#)
- getCameraParamWithProfile() : [ob::Pipeline](#)
- getConfig() : [ob::Pipeline](#)
- getConfigSchema() : [ob::Filter](#)
- getConfigSchemaVec() : [ob::Filter](#)
- getConfigValue() : [ob::Filter](#)
- getConnectionType() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- getCoordinateValueScale() : [ob::PointsFrame](#)
- getCount() : [ob::CameraParamList](#), [ob::DeviceFrameInterleaveList](#), [ob::DeviceList](#),  
[ob::DevicePresetList](#), [ob::FrameSet](#), [ob::OBDepthWorkModeList](#), [ob::OBFilterList](#),  
[ob::PresetResolutionConfigList](#), [ob::SensorList](#), [ob::StreamProfileList](#)
- getCurrentDepthModeName() : [ob::Device](#)
- getCurrentDepthWorkMode() : [ob::Device](#)
- getCurrentPresetName() : [ob::Device](#)
- getD2CDepthProfileList() : [ob::Pipeline](#)
- getData() : [ob::Frame](#)
- getDataSize() : [ob::Frame](#)
- getDepthWorkModeList() : [ob::Device](#)
- getDevice() : [ob::DeviceList](#), [ob::Frame](#), [ob::Pipeline](#)
- getDeviceBySN() : [ob::DeviceList](#)
- getDeviceByUid() : [ob::DeviceList](#)
- getDeviceInfo() : [ob::Device](#)
- getDeviceState() : [ob::Device](#)
- getDeviceType() : [ob::DeviceInfo](#)
- getDiffScaleRange() : [ob::TemporalFilter](#)
- getDispDiffRange() : [ob::NoiseRemovalFilter](#), [ob::SpatialAdvancedFilter](#)
- getDistortion() : [ob::VideoStreamProfile](#)
- getDuration() : [ob::PlaybackDevice](#)
- getEnabledStreamProfileList() : [ob::Config](#)

- `getExceptionType() : ob::Error`
- `getExtensionInfo() : ob::Device`
- `getExtrinsicTo() : ob::StreamProfile`
- `getFilter() : ob::OBFilterList`
- `getFilterMode() : ob::HoleFillingFilter`
- `getFilterParams() : ob::NoiseRemovalFilter, ob::SpatialAdvancedFilter`
- `getFilterVendorSpecificCode() : ob::FilterFactory`
- `getFirmwareVersion() : ob::DeviceInfo`
- `getFloatProperty() : ob::Device`
- `getFloatPropertyRange() : ob::Device`
- `getFormat() : ob::Frame, ob::StreamProfile`
- `getFps() : ob::VideoStreamProfile`
- `getFrame() : ob::FrameSet`
- `getFrameByIndex() : ob::FrameSet`
- `getFullScaleRange() : ob::AccelStreamProfile, ob::GyroStreamProfile`
- `getFunction() : ob::Error`
- `getGlobalTimeStampUs() : ob::Frame`
- `getGyroStreamProfile() : ob::StreamProfileList`
- `getHardwareVersion() : ob::DeviceInfo`
- `getHeight() : ob::PointsFrame, ob::VideoFrame, ob::VideoStreamProfile`
- `getImpl() : ob::Config, ob::Device, ob::Filter, ob::Frame, ob::StreamProfile`
- `getIndex() : ob::Frame`
- `getIntProperty() : ob::Device`
- `getIntPropertyRange() : ob::Device`
- `getIntrinsic() : ob::AccelStreamProfile, ob::GyroStreamProfile, ob::VideoStreamProfile`
- `getIpAddress() : ob::DeviceInfo, ob::DeviceList`
- `getLocalMacAddress() : ob::DeviceList`
- `getMagnitudeRange() : ob::SpatialAdvancedFilter`
- `getMajor() : ob::Version`
- `getMaxRange() : ob::ThresholdFilter`
- `getMaxSizeRange() : ob::NoiseRemovalFilter`
- `getMessage() : ob::Error`
- `getMetadata() : ob::Frame`
- `getMetadataSize() : ob::Frame`
- `getMetadataValue() : ob::Frame`
- `getMinor() : ob::Version`
- `getMinRange() : ob::ThresholdFilter`
- `getMultiDeviceSyncConfig() : ob::Device`
- `getName() : ob::DeviceFrameInterleaveList, ob::DeviceInfo, ob::DeviceList, ob::DevicePresetList, ob::Error, ob::Filter`
- `getOBDepthWorkMode() : ob::OBDepthWorkModeList`
- `getPatch() : ob::Version`
- `getPid() : ob::DeviceInfo, ob::DeviceList`
- `getPixelAvailableBitSize() : ob::VideoFrame`

- `getPixelType() : ob::VideoFrame`
- `getPlaybackStatus() : ob::PlaybackDevice`
- `getPosition() : ob::PlaybackDevice`
- `getPresetResolutionRatioConfig() : ob::PresetResolutionConfigList`
- `getProfile() : ob::StreamProfileList`
- `getRadiusRange() : ob::SpatialAdvancedFilter`
- `getRecommendedFilters() : ob::Sensor`
- `getSampleRate() : ob::AccelStreamProfile, ob::GyroStreamProfile`
- `getScaleRange() : ob::DecimationFilter`
- `getScaleValue() : ob::DecimationFilter`
- `getSelectSequenceId() : ob::SequenceIdFilter`
- `getSensor() : ob::Device, ob::Frame, ob::SensorList`
- `getSensorList() : ob::Device`
- `getSensorType() : ob::SensorList`
- `getSequenceIdList() : ob::SequenceIdFilter`
- `getSequenceIdListSize() : ob::SequenceIdFilter`
- `getSerialNumber() : ob::DeviceInfo, ob::DeviceList`
- `getStageVersion() : ob::Version`
- `getStreamProfile() : ob::Frame`
- `getStreamProfileList() : ob::Pipeline, ob::Sensor`
- `getStructuredData() : ob::Device`
- `getSupportedMinSdkVersion() : ob::DeviceInfo`
- `getSupportedMultiDeviceSyncModeBitmap() : ob::Device`
- `getSupportedProperty() : ob::Device`
- `getSupportedPropertyCount() : ob::Device`
- `getSystemTimeStampUs() : ob::Frame`
- `getTemperature() : ob::AccelFrame, ob::GyroFrame`
- `getTimestampResetConfig() : ob::Device`
- `getTimeStampUs() : ob::Frame`
- `getType() : ob::Frame, ob::Sensor, ob::StreamProfile`
- `getUid() : ob::DeviceInfo, ob::DeviceList`
- `getValue() : ob::AccelFrame, ob::GyroFrame`
- `getValueScale() : ob::DepthFrame`
- `getVersion() : ob::Version`
- `getVid() : ob::DeviceInfo, ob::DeviceList`
- `getVideoStreamProfile() : ob::StreamProfileList`
- `getWeightRange() : ob::TemporalFilter`
- `getWidth() : ob::PointsFrame, ob::VideoFrame, ob::VideoStreamProfile`
- `globalTimeStampUs() : ob::Frame`
- `GyroFrame() : ob::GyroFrame`
- `GyroStreamProfile() : ob::GyroStreamProfile`

Here is a list of all functions with links to the classes they belong to:

- h -

- handle() : [ob::Error](#)
- hardwareVersion() : [ob::DeviceInfo](#)
- hasFrameInterleave() : [ob::DeviceFrameInterleaveList](#)
- hasMetadata() : [ob::Frame](#)
- hasPreset() : [ob::DevicePresetList](#)
- HdrMerge() : [ob::HdrMerge](#)
- height() : [ob::VideoFrame](#), [ob::VideoStreamProfile](#)
- HoleFillingFilter() : [ob::HoleFillingFilter](#)

Here is a list of all functions with links to the classes they belong to:

- i -

- index() : [\*\*ob::Frame\*\*](#)
- init() : [\*\*ob::Filter\*\*](#)
- ipAddress() : [\*\*ob::DeviceInfo\*\*](#), [\*\*ob::DeviceList\*\*](#)
- IRFrame() : [\*\*ob::IRFrame\*\*](#)
- irFrame() : [\*\*ob::FrameSet\*\*](#)
- is() : [\*\*ob::Filter\*\*](#), [\*\*ob::Frame\*\*](#), [\*\*ob::StreamProfile\*\*](#)
- isEnabled() : [\*\*ob::Filter\*\*](#)
- isExtensionInfoExist() : [\*\*ob::Device\*\*](#)
- isFrameInterleaveSupported() : [\*\*ob::Device\*\*](#)
- isGlobalTimestampSupported() : [\*\*ob::Device\*\*](#)
- isPropertySupported() : [\*\*ob::Device\*\*](#)
- isVideoSensorType() : [\*\*ob::TypeHelper\*\*](#)
- isVideoStreamType() : [\*\*ob::TypeHelper\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- I -

- `loadDepthFilterConfig()` : [\*\*ob::Device\*\*](#)
- `loadFrameInterleave()` : [\*\*ob::Device\*\*](#)
- `loadPreset()` : [\*\*ob::Device\*\*](#)
- `loadPresetFromJsonData()` : [\*\*ob::Device\*\*](#)
- `loadPresetFromFile()` : [\*\*ob::Device\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- m -

- metadata() : [ob::Frame](#)
- metadataSize() : [ob::Frame](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- n -

- name() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- NoiseRemovalFilter() : [ob::NoiseRemovalFilter](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- 0 -

- OBDepthWorkModeList() : [\*\*ob::OBDepthWorkModeList\*\*](#)
- OBFILTERList() : [\*\*ob::OBFILTERList\*\*](#)
- operator=() : [\*\*ob::Device\*\*](#), [\*\*ob::PlaybackDevice\*\*](#), [\*\*ob::RecordDevice\*\*](#), [\*\*ob::Sensor\*\*](#), [\*\*ob::StreamProfile\*\*](#)
- operator[](()) : [\*\*ob::OBDepthWorkModeList\*\*](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- p -

- pause() : [ob::PlaybackDevice, ob::RecordDevice](#)
- pid() : [ob::DeviceInfo, ob::DeviceList](#)
- Pipeline() : [ob::Pipeline](#)
- pixelAvailableBitSize() : [ob::VideoFrame](#)
- PlaybackDevice() : [ob::PlaybackDevice](#)
- PointCloudFilter() : [ob::PointCloudFilter](#)
- PointsFrame() : [ob::PointsFrame](#)
- pointsFrame() : [ob::FrameSet](#)
- PresetResolutionConfigList() : [ob::PresetResolutionConfigList](#)
- process() : [ob::Filter](#)
- pushFrame() : [ob::Filter, ob::FrameSet](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- q -

- queryDeviceList() : [\*\*ob::Context\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- r -

- `readCustomerData()` : [\*\*ob::Device\*\*](#)
- `reboot()` : [\*\*ob::Device\*\*](#)
- `RecordDevice()` : [\*\*ob::RecordDevice\*\*](#)
- `reset()` : [\*\*ob::Filter\*\*](#)
- `resume()` : [\*\*ob::PlaybackDevice\*\*](#), [\*\*ob::RecordDevice\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- S -

- sampleRate() : [ob::AccelStreamProfile](#), [ob::GyroStreamProfile](#)
- savePointCloudToPly() : [ob::PointCloudHelper](#)
- seek() : [ob::PlaybackDevice](#)
- selectSequenceId() : [ob::SequenceIdFilter](#)
- sendAndReceiveData() : [ob::Device](#)
- Sensor() : [ob::Sensor](#)
- SensorList() : [ob::SensorList](#)
- SequenceIdFilter() : [ob::SequenceIdFilter](#)
- serialNumber() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- setAlignMode() : [ob::Config](#)
- setAlignToStreamProfile() : [ob::Align](#)
- setBoolProperty() : [ob::Device](#)
- setCallBack() : [ob::Filter](#)
- setCameraParam() : [ob::PointCloudFilter](#)
- setColorDataNormalization() : [ob::PointCloudFilter](#)
- setConfigValue() : [ob::Filter](#)
- setCoordinateDataScaled() : [ob::PointCloudFilter](#)
- setCoordinateSystem() : [ob::PointCloudFilter](#)
- setCreatePointFormat() : [ob::PointCloudFilter](#)
- setDepthScaleRequire() : [ob::Config](#)
- setDeviceChangedCallback() : [ob::Context](#)
- setDeviceStateChangedCallback() : [ob::Device](#)
- setDiffScale() : [ob::TemporalFilter](#)
- setDistortion() : [ob::VideoStreamProfile](#)
- setExtensionsDirectory() : [ob::Context](#)
- setFilterMode() : [ob::HoleFillingFilter](#)
- setFilterParams() : [ob::NoiseRemovalFilter](#), [ob::SpatialAdvancedFilter](#)
- setFloatProperty() : [ob::Device](#)
- setFormatConvertType() : [ob::FormatConvertFilter](#)
- setFrameAggregateOutputMode() : [ob::Config](#)
- setFrameAlignState() : [ob::PointCloudFilter](#)
- setFrameDeviceTimestamp() : [ob::FrameHelper](#)
- setFrameDeviceTimestampUs() : [ob::FrameHelper](#)
- setFrameSystemTimestamp() : [ob::FrameHelper](#)
- setIntProperty() : [ob::Device](#)
- setIntrinsic() : [ob::VideoStreamProfile](#)
- setLoggerSeverity() : [ob::Context](#)
- setLoggerToCallback() : [ob::Context](#)
- setLoggerToConsole() : [ob::Context](#)
- setLoggerToFile() : [ob::Context](#)
- setMatchTargetResolution() : [ob::Align](#)

- setMultiDeviceSyncConfig() : **ob::Device**
- setPlaybackRate() : **ob::PlaybackDevice**
- setPlaybackStatusChangeCallback() : **ob::PlaybackDevice**
- setPositionDataScaled() : **ob::PointCloudFilter**
- setScaleValue() : **ob::DecimationFilter**
- setStructuredData() : **ob::Device**
- setTimestampResetConfig() : **ob::Device**
- setUvcBackendType() : **ob::Context**
- setValueRange() : **ob::ThresholdFilter**
- setWeight() : **ob::TemporalFilter**
- SpatialAdvancedFilter() : **ob::SpatialAdvancedFilter**
- start() : **ob::Pipeline**, **ob::Sensor**
- stop() : **ob::Pipeline**, **ob::Sensor**
- StreamProfile() : **ob::StreamProfile**
- StreamProfileList() : **ob::StreamProfileList**
- supportedMinSdkVersion() : **ob::DeviceInfo**
- switchDepthWorkMode() : **ob::Device**
- switchProfile() : **ob::Sensor**
- systemTimeStamp() : **ob::Frame**
- systemTimeStampUs() : **ob::Frame**

Here is a list of all functions with links to the classes they belong to:

- t -

- temperature() : [\*\*ob::AccelFrame, ob::GyroFrame\*\*](#)
- TemporalFilter() : [\*\*ob::TemporalFilter\*\*](#)
- ThresholdFilter() : [\*\*ob::ThresholdFilter\*\*](#)
- timerSyncWithHost() : [\*\*ob::Device\*\*](#)
- timeStamp() : [\*\*ob::Frame\*\*](#)
- timestampReset() : [\*\*ob::Device\*\*](#)
- timeStampUs() : [\*\*ob::Frame\*\*](#)
- transformation2dto2d() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformation2dto3d() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformation3dto2d() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformation3dto3d() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformationDepthFrameToColorCamera() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformationDepthToPointCloud() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformationDepthToRGBDPointCloud() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- transformationInitXYTables() : [\*\*ob::CoordinateTransformHelper\*\*](#)
- triggerCapture() : [\*\*ob::Device\*\*](#)
- type() : [\*\*ob::Filter, ob::Frame, ob::Sensor, ob::SensorList, ob::StreamProfile\*\*](#)

Here is a list of all functions with links to the classes they belong to:

- u -

- uid() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- updateFirmware() : [ob::Device](#)
- updateFirmwareFromData() : [ob::Device](#)
- updateOptionalDepthPresets() : [ob::Device](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- V -

- value() : [\*\*ob::AccelFrame\*\*](#), [\*\*ob::GyroFrame\*\*](#)
- vid() : [\*\*ob::DeviceInfo\*\*](#), [\*\*ob::DeviceList\*\*](#)
- VideoFrame() : [\*\*ob::VideoFrame\*\*](#)
- VideoStreamProfile() : [\*\*ob::VideoStreamProfile\*\*](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the classes they belong to:

- W -

- waitForFrames() : [ob::Pipeline](#)
- waitForFrameset() : [ob::Pipeline](#)
- what() : [ob::Error](#)
- width() : [ob::VideoFrame](#), [ob::VideoStreamProfile](#)
- writeCustomerData() : [ob::Device](#)

Here is a list of all functions with links to the classes they belong to:

- ~ -

- [~AccelFrame\(\)](#) : **ob::AccelFrame**
- [~AccelStreamProfile\(\)](#) : **ob::AccelStreamProfile**
- [~Align\(\)](#) : **ob::Align**
- [~CameraParamList\(\)](#) : **ob::CameraParamList**
- [~ColorFrame\(\)](#) : **ob::ColorFrame**
- [~ConfidenceFrame\(\)](#) : **ob::ConfidenceFrame**
- [~Config\(\)](#) : **ob::Config**
- [~Context\(\)](#) : **ob::Context**
- [~DecimationFilter\(\)](#) : **ob::DecimationFilter**
- [~DepthFrame\(\)](#) : **ob::DepthFrame**
- [~Device\(\)](#) : **ob::Device**
- [~DeviceFrameInterleaveList\(\)](#) : **ob::DeviceFrameInterleaveList**
- [~DeviceInfo\(\)](#) : **ob::DeviceInfo**
- [~DeviceList\(\)](#) : **ob::DeviceList**
- [~DevicePresetList\(\)](#) : **ob::DevicePresetList**
- [~DisparityTransform\(\)](#) : **ob::DisparityTransform**
- [~Error\(\)](#) : **ob::Error**
- [~Filter\(\)](#) : **ob::Filter**
- [~FormatConvertFilter\(\)](#) : **ob::FormatConvertFilter**
- [~Frame\(\)](#) : **ob::Frame**
- [~FrameSet\(\)](#) : **ob::FrameSet**
- [~GyroFrame\(\)](#) : **ob::GyroFrame**
- [~GyroStreamProfile\(\)](#) : **ob::GyroStreamProfile**
- [~HdrMerge\(\)](#) : **ob::HdrMerge**
- [~HoleFillingFilter\(\)](#) : **ob::HoleFillingFilter**
- [~IRFrame\(\)](#) : **ob::IRFrame**
- [~NoiseRemovalFilter\(\)](#) : **ob::NoiseRemovalFilter**
- [~OBDepthWorkModeList\(\)](#) : **ob::OBDepthWorkModeList**
- [~OBFilterList\(\)](#) : **ob::OBFilterList**
- [~Pipeline\(\)](#) : **ob::Pipeline**
- [~PlaybackDevice\(\)](#) : **ob::PlaybackDevice**
- [~PointCloudFilter\(\)](#) : **ob::PointCloudFilter**
- [~PointsFrame\(\)](#) : **ob::PointsFrame**
- [~PresetResolutionConfigList\(\)](#) : **ob::PresetResolutionConfigList**
- [~RecordDevice\(\)](#) : **ob::RecordDevice**
- [~Sensor\(\)](#) : **ob::Sensor**
- [~SensorList\(\)](#) : **ob::SensorList**
- [~SequenceIdFilter\(\)](#) : **ob::SequenceIdFilter**
- [~SpatialAdvancedFilter\(\)](#) : **ob::SpatialAdvancedFilter**
- [~StreamProfile\(\)](#) : **ob::StreamProfile**
- [~StreamProfileList\(\)](#) : **ob::StreamProfileList**

- ~TemporalFilter() : **ob::TemporalFilter**
- ~ThresholdFilter() : **ob::ThresholdFilter**
- ~VideoFrame() : **ob::VideoFrame**
- ~VideoStreamProfile() : **ob::VideoStreamProfile**

Generated on for OrbbeeSDK by **doxygen** 1.14.0

Here is a list of all class members with links to the classes they belong to:

- g -

- g : [OBColorPoint](#)
- gain\_1 : [HDR\\_CONFIG](#)
- gain\_2 : [HDR\\_CONFIG](#)
- gateway : [OBNetIpConfig](#)
- getAccelStreamProfile() : [ob::StreamProfileList](#)
- getAlignToStreamType() : [ob::Align](#)
- getAlphaRange() : [ob::SpatialAdvancedFilter](#)
- getArgs() : [ob::Error](#)
- getAsicName() : [ob::DeviceInfo](#)
- getAvailableFrameInterleaveList() : [ob::Device](#)
- getAvailablePresetList() : [ob::Device](#)
- getAvailablePresetResolutionConfigeList() : [ob::Device](#)
- getBoolProperty() : [ob::Device](#)
- getBoolPropertyRange() : [ob::Device](#)
- getCalibrationCameraParamList() : [ob::Device](#)
- getCalibrationParam() : [ob::Pipeline](#)
- getCameraParam() : [ob::CameraParamList](#), [ob::Pipeline](#)
- getCameraParamWithProfile() : [ob::Pipeline](#)
- getConfig() : [ob::Pipeline](#)
- getConfigSchema() : [ob::Filter](#)
- getConfigSchemaVec() : [ob::Filter](#)
- getConfigValue() : [ob::Filter](#)
- getConnectionType() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- getCoordinateValueScale() : [ob::PointsFrame](#)
- getCount() : [ob::CameraParamList](#), [ob::DeviceFrameInterleaveList](#), [ob::DeviceList](#),  
[ob::DevicePresetList](#), [ob::FrameSet](#), [ob::OBDepthWorkModeList](#), [ob::OBFilterList](#),  
[ob::PresetResolutionConfigeList](#), [ob::SensorList](#), [ob::StreamProfileList](#)
- getCurrentDepthModeName() : [ob::Device](#)
- getCurrentDepthWorkMode() : [ob::Device](#)
- getCurrentPresetName() : [ob::Device](#)
- getD2CDepthProfileList() : [ob::Pipeline](#)
- getData() : [ob::Frame](#)
- getDataSize() : [ob::Frame](#)
- getDepthWorkModeList() : [ob::Device](#)
- getDevice() : [ob::DeviceList](#), [ob::Frame](#), [ob::Pipeline](#)
- getDeviceBySN() : [ob::DeviceList](#)
- getDeviceByUid() : [ob::DeviceList](#)
- getDeviceInfo() : [ob::Device](#)
- getDeviceState() : [ob::Device](#)
- getDeviceType() : [ob::DeviceInfo](#)
- getDiffScaleRange() : [ob::TemporalFilter](#)

- getDispDiffRange() : **ob::NoiseRemovalFilter, ob::SpatialAdvancedFilter**
- getDistortion() : **ob::VideoStreamProfile**
- getDuration() : **ob::PlaybackDevice**
- getEnabledStreamProfileList() : **ob::Config**
- getExceptionType() : **ob::Error**
- getExtensionInfo() : **ob::Device**
- getExtrinsicTo() : **ob::StreamProfile**
- getFilter() : **ob::OBFilterList**
- getFilterMode() : **ob::HoleFillingFilter**
- getFilterParams() : **ob::NoiseRemovalFilter, ob::SpatialAdvancedFilter**
- getFilterVendorSpecificCode() : **ob::FilterFactory**
- getFirmwareVersion() : **ob::DeviceInfo**
- getFloatProperty() : **ob::Device**
- getFloatPropertyRange() : **ob::Device**
- getFormat() : **ob::Frame, ob::StreamProfile**
- getFps() : **ob::VideoStreamProfile**
- getFrame() : **ob::FrameSet**
- getFrameByIndex() : **ob::FrameSet**
- getFullScaleRange() : **ob::AccelStreamProfile, ob::GyroStreamProfile**
- getFunction() : **ob::Error**
- getGlobalTimeStampUs() : **ob::Frame**
- getGyroStreamProfile() : **ob::StreamProfileList**
- getHardwareVersion() : **ob::DeviceInfo**
- getHeight() : **ob::PointsFrame, ob::VideoFrame, ob::VideoStreamProfile**
- getImpl() : **ob::Config, ob::Device, ob::Filter, ob::Frame, ob::StreamProfile**
- getIndex() : **ob::Frame**
- getIntProperty() : **ob::Device**
- getIntPropertyRange() : **ob::Device**
- getIntrinsic() : **ob::AccelStreamProfile, ob::GyroStreamProfile, ob::VideoStreamProfile**
- getIpAddress() : **ob::DeviceInfo, ob::DeviceList**
- getLocalMacAddress() : **ob::DeviceList**
- getMagnitudeRange() : **ob::SpatialAdvancedFilter**
- getMajor() : **ob::Version**
- getMaxRange() : **ob::ThresholdFilter**
- getMaxSizeRange() : **ob::NoiseRemovalFilter**
- getMessage() : **ob::Error**
- getMetadata() : **ob::Frame**
- getMetadataSize() : **ob::Frame**
- getMetadataValue() : **ob::Frame**
- getMinor() : **ob::Version**
- getMinRange() : **ob::ThresholdFilter**
- getMultiDeviceSyncConfig() : **ob::Device**
- getName() : **ob::DeviceFrameInterleaveList, ob::DeviceInfo, ob::DeviceList, ob::DevicePresetList, ob::Error, ob::Filter**

- getOBDepthWorkMode() : **ob::OBDepthWorkModeList**
- getPatch() : **ob::Version**
- getPid() : **ob::DeviceInfo, ob::DeviceList**
- getPixelAvailableBitSize() : **ob::VideoFrame**
- getPixelType() : **ob::VideoFrame**
- getPlaybackStatus() : **ob::PlaybackDevice**
- getPosition() : **ob::PlaybackDevice**
- getPresetResolutionRatioConfig() : **ob::PresetResolutionConfigList**
- getProfile() : **ob::StreamProfileList**
- getRadiusRange() : **ob::SpatialAdvancedFilter**
- getRecommendedFilters() : **ob::Sensor**
- getSampleRate() : **ob::AccelStreamProfile, ob::GyroStreamProfile**
- getScaleRange() : **ob::DecimationFilter**
- getScaleValue() : **ob::DecimationFilter**
- getSelectSequenceId() : **ob::SequenceIdFilter**
- getSensor() : **ob::Device, ob::Frame, ob::SensorList**
- getSensorList() : **ob::Device**
- getSensorType() : **ob::SensorList**
- getSequenceIdList() : **ob::SequenceIdFilter**
- getSequenceIdListSize() : **ob::SequenceIdFilter**
- getSerialNumber() : **ob::DeviceInfo, ob::DeviceList**
- getStageVersion() : **ob::Version**
- getStreamProfile() : **ob::Frame**
- getStreamProfileList() : **ob::Pipeline, ob::Sensor**
- getStructuredData() : **ob::Device**
- getSupportedMinSdkVersion() : **ob::DeviceInfo**
- getSupportedMultiDeviceSyncModeBitmap() : **ob::Device**
- getSupportedProperty() : **ob::Device**
- getSupportedPropertyCount() : **ob::Device**
- getSystemTimeStampUs() : **ob::Frame**
- getTemperature() : **ob::AccelFrame, ob::GyroFrame**
- getTimestampResetConfig() : **ob::Device**
- getTimeStampUs() : **ob::Frame**
- getType() : **ob::Frame, ob::Sensor, ob::StreamProfile**
- getUid() : **ob::DeviceInfo, ob::DeviceList**
- getValue() : **ob::AccelFrame, ob::GyroFrame**
- getValueScale() : **ob::DepthFrame**
- getVersion() : **ob::Version**
- getVid() : **ob::DeviceInfo, ob::DeviceList**
- getVideoStreamProfile() : **ob::StreamProfileList**
- getWeightRange() : **ob::TemporalFilter**
- getWidth() : **ob::PointsFrame, ob::VideoFrame, ob::VideoStreamProfile**
- globalTimeStampUs() : **ob::Frame**
- gravity : **OBAccelIntrinsic**

- GyroFrame() : [\*\*ob::GyroFrame\*\*](#)
- GyroStreamProfile() : [\*\*ob::GyroStreamProfile\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- h -

- handle() : [ob::Error](#)
- hardwareVersion() : [ob::DeviceInfo](#)
- hasFrameInterleave() : [ob::DeviceFrameInterleaveList](#)
- hasMetadata() : [ob::Frame](#)
- hasPreset() : [ob::DevicePresetList](#)
- HdrMerge() : [ob::HdrMerge](#)
- height() : [ob::VideoFrame](#), [ob::VideoStreamProfile](#), [ob\\_margin\\_filter\\_config](#),  
[OBCameraIntrinsic](#), [OBMGCFilterConfig](#), [OBPresetResolutionConfig](#), [OBRect](#), [OBXYTables](#)
- HoleFillingFilter() : [ob::HoleFillingFilter](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- i -

- id : [OBPropertyItem](#)
- impl\_ : [ob::Device](#), [ob::Filter](#), [ob::Frame](#), [ob::Sensor](#), [ob::StreamProfile](#), [ob::StreamProfileList](#)
- imuTemp : [OBDeviceTemperature](#)
- index() : [ob::Frame](#)
- init() : [ob::Filter](#)
- intrinsics : [OBCalibrationParam](#)
- invalidDisp : [OBDisparityParam](#)
- ipAddress() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- irDecimationFactor : [OBPresetResolutionConfig](#)
- IRFrame() : [ob::IRFrame](#)
- irFrame() : [ob::FrameSet](#)
- irLeftTemp : [OBDeviceTemperature](#)
- irRightTemp : [OBDeviceTemperature](#)
- irTemp : [OBDeviceTemperature](#)
- irTriggerSignalInDelay : [OBDeviceSyncConfig](#)
- is() : [ob::Filter](#), [ob::Frame](#), [ob::StreamProfile](#)
- isDualCamera : [OBDisparityParam](#)
- isEnabled() : [ob::Filter](#)
- isExtensionInfoExist() : [ob::Device](#)
- isFrameInterleaveSupported() : [ob::Device](#)
- isGlobalTimestampSupported() : [ob::Device](#)
- isMirrored : [OBCameraParam](#)
- isPropertySupported() : [ob::Device](#)
- isVideoSensorType() : [ob::TypeHelper](#)
- isVideoStreamType() : [ob::TypeHelper](#)

Here is a list of all class members with links to the classes they belong to:

- k -

- k1 : [OBCameraDistortion](#)
- k2 : [OBCameraDistortion](#)
- k3 : [OBCameraDistortion](#)
- k4 : [OBCameraDistortion](#)
- k5 : [OBCameraDistortion](#)
- k6 : [OBCameraDistortion](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- I -

- ldmTemp : [\*\*OBDeviceTemperature\*\*](#)
- limit\_x\_th : [\*\*ob\\_margin\\_filter\\_config\*\*](#), [\*\*OBMGCFilterConfig\*\*](#)
- limit\_y\_th : [\*\*ob\\_margin\\_filter\\_config\*\*](#), [\*\*OBMGCFilterConfig\*\*](#)
- loadDepthFilterConfig() : [\*\*ob::Device\*\*](#)
- loadFrameInterleave() : [\*\*ob::Device\*\*](#)
- loadPreset() : [\*\*ob::Device\*\*](#)
- loadPresetFromJsonData() : [\*\*ob::Device\*\*](#)
- loadPresetFromFile() : [\*\*ob::Device\*\*](#)
- LogCallback : [\*\*ob::Context\*\*](#)
- lower : [\*\*OBTofExposureThresholdControl\*\*](#)

Here is a list of all class members with links to the classes they belong to:

- m -

- magnitude : [OBSpatialAdvancedFilterParams](#)
- mainBoardTemp : [OBDeviceTemperature](#)
- major : [OBProtocolVersion](#)
- margin\_x\_th : [ob\\_margin\\_filter\\_config](#), [OBMGCFilterConfig](#)
- margin\_y\_th : [ob\\_margin\\_filter\\_config](#), [OBMGCFilterConfig](#)
- marginBottomTh : [OBEdgeNoiseRemovalFilterParams](#)
- marginLeftTh : [OBEdgeNoiseRemovalFilterParams](#)
- marginRightTh : [OBEdgeNoiseRemovalFilterParams](#)
- marginTopTh : [OBEdgeNoiseRemovalFilterParams](#)
- mask : [OBNetIpConfig](#)
- max : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#),  
[OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- max\_radius : [OBMGCFilterConfig](#)
- max\_size : [OBNoiseRemovalFilterParams](#)
- max\_width\_left : [OBMGCFilterConfig](#)
- max\_width\_right : [OBMGCFilterConfig](#)
- mcuTriggerFrequency : [OBDeviceSyncConfig](#)
- message : [ob\\_error](#)
- metadata() : [ob::Frame](#)
- metadataSize() : [ob::Frame](#)
- min : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#),  
[OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- minDisparity : [OBDisparityParam](#)
- minor : [OBProtocolVersion](#)
- model : [OBCameraDistortion](#)

Here is a list of all class members with links to the classes they belong to:

- n -

- name() : [ob::DeviceInfo](#), [ob::DeviceList](#), [OBDepthWorkMode](#), [OBFilterConfigSchemaItem](#), [OBPropertyItem](#), [OBSequenceIdItem](#)
- name\_ : [ob::Filter](#)
- noiseDensity : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- NoiseRemovalFilter() : [ob::NoiseRemovalFilter](#)
- numberStr : [OBDeviceSerialNumber](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- 0 -

- OBDepthWorkModeList() : [ob::OBDepthWorkModeList](#)
- OBFILTERList() : [ob::OBFILTERList](#)
- offset : [OBDataChunk](#)
- offset0 : [OBDispOffsetConfig](#)
- offset1 : [OBDispOffsetConfig](#)
- operator=( ) : [ob::Device](#), [ob::PlaybackDevice](#), [ob::RecordDevice](#), [ob::Sensor](#), [ob::StreamProfile](#)
- operator[]( ) : [ob::OBDepthWorkModeList](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- p -

- p1 : [OBCameraDistortion](#)
- p2 : [OBCameraDistortion](#)
- packMode : [OBDisparityParam](#)
- patch : [OBProtocolVersion](#)
- pause() : [ob::PlaybackDevice](#), [ob::RecordDevice](#)
- permission : [OBPropertyItem](#)
- pid() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- Pipeline() : [ob::Pipeline](#)
- pixelAvailableBitSize() : [ob::VideoFrame](#)
- PlaybackDevice() : [ob::PlaybackDevice](#)
- PointCloudFilter() : [ob::PointCloudFilter](#)
- PointsFrame() : [ob::PointsFrame](#)
- pointsFrame() : [ob::FrameSet](#)
- PresetResolutionConfigList() : [ob::PresetResolutionConfigList](#)
- process() : [ob::Filter](#)
- pushFrame() : [ob::Filter](#), [ob::FrameSet](#)

Here is a list of all class members with links to the classes they belong to:

- q -

- queryDeviceList() : [ob::Context](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- r -

- r : [OBColorPoint](#)
- radius : [OBSpatialAdvancedFilterParams](#)
- randomWalk : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- readCustomerData() : [ob::Device](#)
- reboot() : [ob::Device](#)
- RecordDevice() : [ob::RecordDevice](#)
- referenceTemp : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- reserved : [OBDispOffsetConfig](#)
- reset() : [ob::Filter](#)
- resume() : [ob::PlaybackDevice](#), [ob::RecordDevice](#)
- rgbDistortion : [OBCameraParam](#)
- rgbIntrinsic : [OBCameraParam](#)
- rgbTemp : [OBDeviceTemperature](#)
- rgbTriggerSignalInDelay : [OBDeviceSyncConfig](#)
- rot : [OBD2CTransform](#)

Here is a list of all class members with links to the classes they belong to:

- S -

- sampleRate() : [ob::AccelStreamProfile](#), [ob::GyroStreamProfile](#)
- savePointCloudToPly() : [ob::PointCloudHelper](#)
- scaleMisalignment : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- seek() : [ob::PlaybackDevice](#)
- selectSequenceId() : [ob::SequenceIdFilter](#)
- sendAndReceiveData() : [ob::Device](#)
- Sensor() : [ob::Sensor](#)
- SensorList() : [ob::SensorList](#)
- sequence\_name : [HDR\\_CONFIG](#)
- SequenceIdFilter() : [ob::SequenceIdFilter](#)
- sequenceSelectId : [OBSequenceIdItem](#)
- serialNumber() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- setAlignMode() : [ob::Config](#)
- setAlignToStreamProfile() : [ob::Align](#)
- setBoolProperty() : [ob::Device](#)
- setCallBack() : [ob::Filter](#)
- setCameraParam() : [ob::PointCloudFilter](#)
- setColorDataNormalization() : [ob::PointCloudFilter](#)
- setConfigValue() : [ob::Filter](#)
- setCoordinateDataScaled() : [ob::PointCloudFilter](#)
- setCoordinateSystem() : [ob::PointCloudFilter](#)
- setCreatePointFormat() : [ob::PointCloudFilter](#)
- setDepthScaleRequire() : [ob::Config](#)
- setDeviceChangedCallback() : [ob::Context](#)
- setDeviceStateChangedCallback() : [ob::Device](#)
- setDiffScale() : [ob::TemporalFilter](#)
- setDistortion() : [ob::VideoStreamProfile](#)
- setExtensionsDirectory() : [ob::Context](#)
- setFilterMode() : [ob::HoleFillingFilter](#)
- setFilterParams() : [ob::NoiseRemovalFilter](#), [ob::SpatialAdvancedFilter](#)
- setFloatProperty() : [ob::Device](#)
- setFormatConvertType() : [ob::FormatConvertFilter](#)
- setFrameAggregateOutputMode() : [ob::Config](#)
- setFrameAlignState() : [ob::PointCloudFilter](#)
- setFrameDeviceTimestamp() : [ob::FrameHelper](#)
- setFrameDeviceTimestampUs() : [ob::FrameHelper](#)
- setFrameSystemTimestamp() : [ob::FrameHelper](#)
- setIntProperty() : [ob::Device](#)
- setIntrinsic() : [ob::VideoStreamProfile](#)
- setLoggerSeverity() : [ob::Context](#)
- setLoggerToCallback() : [ob::Context](#)

- setLoggerToConsole() : **ob::Context**
- setLoggerToFile() : **ob::Context**
- setMatchTargetResolution() : **ob::Align**
- setMultiDeviceSyncConfig() : **ob::Device**
- setPlaybackRate() : **ob::PlaybackDevice**
- setPlaybackStatusChangeCallback() : **ob::PlaybackDevice**
- setPositionDataScaled() : **ob::PointCloudFilter**
- setScaleValue() : **ob::DecimationFilter**
- setStructuredData() : **ob::Device**
- setTimestampResetConfig() : **ob::Device**
- setUvcBackendType() : **ob::Context**
- setValueRange() : **ob::ThresholdFilter**
- setWeight() : **ob::TemporalFilter**
- size : **OBDataChunk**
- SpatialAdvancedFilter() : **ob::SpatialAdvancedFilter**
- start() : **ob::Pipeline**, **ob::Sensor**
- status : **ob\_error**
- step : **OBBoolPropertyRange**, **OBFilterConfigSchemaItem**, **OBFloatPropertyRange**,  
**OBIntPropertyRange**, **OBUInt16PropertyRange**, **OBUInt8PropertyRange**
- stop() : **ob::Pipeline**, **ob::Sensor**
- StreamProfile() : **ob::StreamProfile**
- StreamProfileList() : **ob::StreamProfileList**
- supportedMinSdkVersion() : **ob::DeviceInfo**
- switchDepthWorkMode() : **ob::Device**
- switchProfile() : **ob::Sensor**
- syncMode : **ob\_multi\_device\_sync\_config**, **OBDeviceSyncConfig**
- systemTimeStamp() : **ob::Frame**
- systemTimeStampUs() : **ob::Frame**

Here is a list of all class members with links to the classes they belong to:

- t -

- tag : [OBDepthWorkMode](#)
- tecTemp : [OBDeviceTemperature](#)
- temperature() : [ob::AccelFrame](#), [ob::GyroFrame](#)
- TemporalFilter() : [ob::TemporalFilter](#)
- tempSlope : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- threshold : [OBCompressionParams](#)
- ThresholdFilter() : [ob::ThresholdFilter](#)
- timerSyncWithHost() : [ob::Device](#)
- timeStamp() : [ob::Frame](#)
- timestamp\_reset\_delay\_us : [ob\\_device\\_timestamp\\_reset\\_config](#)
- timestamp\_reset\_signal\_output\_enable : [ob\\_device\\_timestamp\\_reset\\_config](#)
- timestampReset() : [ob::Device](#)
- timeStampUs() : [ob::Frame](#)
- trans : [OBD2CTransform](#)
- transform : [OBCameraParam](#)
- transformation2dto2d() : [ob::CoordinateTransformHelper](#)
- transformation2dto3d() : [ob::CoordinateTransformHelper](#)
- transformation3dto2d() : [ob::CoordinateTransformHelper](#)
- transformation3dto3d() : [ob::CoordinateTransformHelper](#)
- transformationDepthFrameToColorCamera() : [ob::CoordinateTransformHelper](#)
- transformationDepthToPointCloud() : [ob::CoordinateTransformHelper](#)
- transformationDepthToRGBDPointCloud() : [ob::CoordinateTransformHelper](#)
- transformationInitXYTables() : [ob::CoordinateTransformHelper](#)
- trigger2ImageDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- triggerCapture() : [ob::Device](#)
- triggerOutDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- triggerOutEnable : [ob\\_multi\\_device\\_sync\\_config](#)
- type() : [ob::Filter](#), [ob::Frame](#), [ob::Sensor](#), [ob::SensorList](#), [ob::StreamProfile](#),  
[OBEdgeNoiseRemovalFilterParams](#), [OBFilterConfigSchemaItem](#),  
[OBNoiseRemovalFilterParams](#), [OBPropertyItem](#)

Here is a list of all typedefs with links to the classes they belong to:

- BufferDestroyCallback : [\*\*ob::FrameFactory\*\*](#)
- DeviceChangedCallback : [\*\*ob::Context\*\*](#)
- DeviceFwUpdateCallback : [\*\*ob::Device\*\*](#)
- DeviceStateChangedCallback : [\*\*ob::Device\*\*](#)
- FrameCallback : [\*\*ob::Sensor\*\*](#)
- FrameSetCallback : [\*\*ob::Pipeline\*\*](#)
- LogCallback : [\*\*ob::Context\*\*](#)
- valueType : [\*\*ob::RangeTraits< T >\*\*](#), [\*\*ob::RangeTraits< OBFloatPropertyRange >\*\*](#),  
[\*\*ob::RangeTraits< OBIntPropertyRange >\*\*](#), [\*\*ob::RangeTraits< OBUInt16PropertyRange >\*\*](#),  
[\*\*ob::RangeTraits< OBUInt8PropertyRange >\*\*](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- u -

- uid() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- unit : [OBDisparityParam](#)
- updateFirmware() : [ob::Device](#)
- updateFirmwareFromData() : [ob::Device](#)
- updateOptionalDepthPresets() : [ob::Device](#)
- upper : [OBTofExposureThresholdControl](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- V -

- value() : [ob::AccelFrame](#), [ob::GyroFrame](#)
- valueType : [ob::RangeTraits< T >](#), [ob::RangeTraits< OBFloatPropertyRange >](#),  
[ob::RangeTraits< OBIntPropertyRange >](#), [ob::RangeTraits< OB UInt16PropertyRange >](#),  
[ob::RangeTraits< OB UInt8PropertyRange >](#)
- vid() : [ob::DeviceInfo](#), [ob::DeviceList](#)
- VideoFrame() : [ob::VideoFrame](#)
- VideoStreamProfile() : [ob::VideoStreamProfile](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- a -

- address : [OBNetIpConfig](#)
- alpha : [OBSpatialAdvancedFilterParams](#)
- args : [ob\\_error](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- b -

- b : [OBColorPoint](#)
- baseline : [BASELINE\\_CALIBRATION\\_PARAM](#), [OBDisparityParam](#)
- bias : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- bitSize : [OBDisparityParam](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- c -

- callback\_ : [ob::Filter](#), [ob::Sensor](#)
- checksum : [OBDepthWorkMode](#)
- chipBottomTemp : [OBDeviceTemperature](#)
- chipTopTemp : [OBDeviceTemperature](#)
- colorDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- configSchemaVec\_ : [ob::Filter](#)
- cpuTemp : [OBDeviceTemperature](#)
- cur : [OBBoolPropertyRange](#), [OBFloatPropertyRange](#), [OBIntPropertyRange](#),  
[OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- cx : [OBCameraIntrinsic](#)
- cy : [OBCameraIntrinsic](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- d -

- data : [OBDataChunk](#)
- def : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#), [OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- depthDecimationFactor : [OBPresetResolutionConfig](#)
- depthDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- depthDistortion : [OBCameraParam](#)
- depthIntrinsic : [OBCameraParam](#)
- desc : [OBFilterConfigSchemaItem](#)
- deviceId : [OBDeviceSyncConfig](#)
- deviceStateChangeCallback\_ : [ob::Device](#)
- deviceTriggerSignalOutDelay : [OBDeviceSyncConfig](#)
- deviceTriggerSignalOutPolarity : [OBDeviceSyncConfig](#)
- dhcp : [OBNetIpConfig](#)
- disp\_diff : [OBNoiseRemovalFilterParams](#), [OBSpatialAdvancedFilterParams](#)
- dispIntPlace : [OBDisparityParam](#)
- dispOffset : [OBDisparityParam](#)
- distortion : [OBCalibrationParam](#)

Here is a list of all variables with links to the classes they belong to:

- e -

- enable : [HDR\\_CONFIG](#), [ob\\_device\\_timestamp\\_reset\\_config](#), [OBDispOffsetConfig](#)
- enable\_direction : [ob\\_margin\\_filter\\_config](#)
- exception\_type : [ob\\_error](#)
- exposure\_1 : [HDR\\_CONFIG](#)
- exposure\_2 : [HDR\\_CONFIG](#)
- extrinsics : [OBCalibrationParam](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- f -

- framesPerTrigger : [\*\*ob\\_multi\\_device\\_sync\\_config\*\*](#)
- fullDataSize : [\*\*OBDataChunk\*\*](#)
- function : [\*\*ob\\_error\*\*](#)
- fwUpdateCallback\_ : [\*\*ob::Device\*\*](#)
- fx : [\*\*OBCameraIntrinsic\*\*](#), [\*\*OBDisparityParam\*\*](#)
- fy : [\*\*OBCameraIntrinsic\*\*](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- g -

- g : [OBColorPoint](#)
- gain\_1 : [HDR\\_CONFIG](#)
- gain\_2 : [HDR\\_CONFIG](#)
- gateway : [OBNetIpConfig](#)
- gravity : [OBAccelIntrinsic](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- h -

- height : [ob\\_margin\\_filter\\_config](#), [OBCameraIntrinsic](#), [OBMGCFilterConfig](#),  
[OBPresetResolutionConfig](#), [OBRect](#), [OBXYTables](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- i -

- id : [OBPropertyItem](#)
- impl\_ : [ob::Device](#), [ob::Filter](#), [ob::Frame](#), [ob::Sensor](#), [ob::StreamProfile](#), [ob::StreamProfileList](#)
- imuTemp : [OBDeviceTemperature](#)
- intrinsics : [OBCalibrationParam](#)
- invalidDisp : [OBDisparityParam](#)
- irDecimationFactor : [OBPresetResolutionConfig](#)
- irLeftTemp : [OBDeviceTemperature](#)
- irRightTemp : [OBDeviceTemperature](#)
- irTemp : [OBDeviceTemperature](#)
- irTriggerSignalInDelay : [OBDeviceSyncConfig](#)
- isDualCamera : [OBDisparityParam](#)
- isMirrored : [OBCameraParam](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- k -

- k1 : [OBCameraDistortion](#)
- k2 : [OBCameraDistortion](#)
- k3 : [OBCameraDistortion](#)
- k4 : [OBCameraDistortion](#)
- k5 : [OBCameraDistortion](#)
- k6 : [OBCameraDistortion](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- 1 -

- ldmTemp : [OBDeviceTemperature](#)
- limit\_x\_th : [ob\\_margin\\_filter\\_config](#), [OBMGCFilterConfig](#)
- limit\_y\_th : [ob\\_margin\\_filter\\_config](#), [OBMGCFilterConfig](#)
- lower : [OBTofExposureThresholdControl](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- m -

- magnitude : [OBSpatialAdvancedFilterParams](#)
- mainBoardTemp : [OBDeviceTemperature](#)
- major : [OBProtocolVersion](#)
- margin\_x\_th : [ob\\_margin\\_filter\\_config](#), [OBMGCFilterConfig](#)
- margin\_y\_th : [ob\\_margin\\_filter\\_config](#), [OBMGCFilterConfig](#)
- marginBottomTh : [OBEdgeNoiseRemovalFilterParams](#)
- marginLeftTh : [OBEdgeNoiseRemovalFilterParams](#)
- marginRightTh : [OBEdgeNoiseRemovalFilterParams](#)
- marginTopTh : [OBEdgeNoiseRemovalFilterParams](#)
- mask : [OBNetIpConfig](#)
- max : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#),  
[OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- max\_radius : [OBMGCFilterConfig](#)
- max\_size : [OBNoiseRemovalFilterParams](#)
- max\_width\_left : [OBMGCFilterConfig](#)
- max\_width\_right : [OBMGCFilterConfig](#)
- mcuTriggerFrequency : [OBDeviceSyncConfig](#)
- message : [ob\\_error](#)
- min : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#),  
[OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- minDisparity : [OBDisparityParam](#)
- minor : [OBProtocolVersion](#)
- model : [OBCameraDistortion](#)

Here is a list of all variables with links to the classes they belong to:

- n -

- name : [OBDepthWorkMode](#), [OBFilterConfigSchemaItem](#), [OBPropertyItem](#), [OBSequenceIdItem](#)
- name\_ : [ob::Filter](#)
- noiseDensity : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- numberStr : [OBDeviceSerialNumber](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- 0 -

- offset : [OBDataChunk](#)
- offset0 : [OBDispOffsetConfig](#)
- offset1 : [OBDispOffsetConfig](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- p -

- p1 : [OBCameraDistortion](#)
- p2 : [OBCameraDistortion](#)
- packMode : [OBDisparityParam](#)
- patch : [OBProtocolVersion](#)
- permission : [OBPropertyItem](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- r -

- r : [OBColorPoint](#)
- radius : [OBSpatialAdvancedFilterParams](#)
- randomWalk : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- referenceTemp : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- reserved : [OBDispOffsetConfig](#)
- rgbDistortion : [OBCameraParam](#)
- rgbIntrinsic : [OBCameraParam](#)
- rgbTemp : [OBDeviceTemperature](#)
- rgbTriggerSignalInDelay : [OBDeviceSyncConfig](#)
- rot : [OBD2CTransform](#)

Here is a list of all variables with links to the classes they belong to:

- S -

- scaleMisalignment : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- sequence\_name : [HDR\\_CONFIG](#)
- sequenceSelectId : [OBSequenceIdItem](#)
- size : [OBDataChunk](#)
- status : [ob\\_error](#)
- step : [OBBoolPropertyRange](#), [OBFilterConfigSchemaItem](#), [OBFloatPropertyRange](#),  
[OBIntPropertyRange](#), [OBUInt16PropertyRange](#), [OBUInt8PropertyRange](#)
- syncMode : [ob\\_multi\\_device\\_sync\\_config](#), [OBDeviceSyncConfig](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- t -

- tag : [OBDepthWorkMode](#)
- tecTemp : [OBDeviceTemperature](#)
- tempSlope : [OBAccelIntrinsic](#), [OBGyroIntrinsic](#)
- threshold : [OBCompressionParams](#)
- timestamp\_reset\_delay\_us : [ob\\_device\\_timestamp\\_reset\\_config](#)
- timestamp\_reset\_signal\_output\_enable : [ob\\_device\\_timestamp\\_reset\\_config](#)
- trans : [OBD2CTransform](#)
- transform : [OBCameraParam](#)
- trigger2ImageDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- triggerOutDelayUs : [ob\\_multi\\_device\\_sync\\_config](#)
- triggerOutEnable : [ob\\_multi\\_device\\_sync\\_config](#)
- type : [OBEdgeNoiseRemovalFilterParams](#), [OBFilterConfigSchemaItem](#),  
[OBNoiseRemovalFilterParams](#), [OBPropertyItem](#)

Here is a list of all variables with links to the classes they belong to:

- u -

- unit : [OBDisparityParam](#)
- upper : [OBTofExposureThresholdControl](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- W -

- width : [ob\\_margin\\_filter\\_config](#), [OBCameraIntrinsic](#), [OBMGCFConfig](#),  
[OBPresetResolutionConfig](#), [OBRect](#), [OBXYTables](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all variables with links to the classes they belong to:

- X -

- x : [OBAccelValue](#), [OBColorPoint](#), [OBPoint2f](#), [OBPoint](#), [OBRect](#)
- x0\_left : [AE\\_ROI](#)
- x1\_right : [AE\\_ROI](#)
- xTable : [OBXYTables](#)

Here is a list of all variables with links to the classes they belong to:

- y -

- y : [OBAccelValue](#), [OBColorPoint](#), [OBPoint2f](#), [OBPoint](#), [OBRect](#)
- y0\_top : [AE\\_ROI](#)
- y1\_bottom : [AE\\_ROI](#)
- yTable : [OBXYTables](#)

Here is a list of all variables with links to the classes they belong to:

- Z -

- z : [OBAccelValue](#), [OBColorPoint](#), [OBPoint](#)
- zpd : [BASELINE\\_CALIBRATION\\_PARAM](#), [OBDisparityParam](#)
- zpps : [OBDisparityParam](#)

Here is a list of all class members with links to the classes they belong to:

- W -

- waitForFrames() : [ob::Pipeline](#)
- waitForFrameset() : [ob::Pipeline](#)
- what() : [ob::Error](#)
- width() : [ob::VideoFrame](#), [ob::VideoStreamProfile](#), [ob\\_margin\\_filter\\_config](#),  
[OBCameraIntrinsic](#), [OBMGCFilterConfig](#), [OBPresetResolutionConfig](#), [OBRect](#), [OBXYTables](#)
- writeCustomerData() : [ob::Device](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- X -

- x : [OBAccelValue](#), [OBColorPoint](#), [OBPoint2f](#), [OBPoint](#), [OBRect](#)
- x0\_left : [AE\\_ROI](#)
- x1\_right : [AE\\_ROI](#)
- xTable : [OBXYTables](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- y -

- y : [OBAccelValue](#), [OBColorPoint](#), [OBPoint2f](#), [OBPoint](#), [OBRect](#)
- y0\_top : [AE\\_ROI](#)
- y1\_bottom : [AE\\_ROI](#)
- yTable : [OBXYTables](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- Z -

- z : [OBAccelValue](#), [OBColorPoint](#), [OBPoint](#)
- zpd : [BASELINE\\_CALIBRATION\\_PARAM](#), [OBDisparityParam](#)
- zpps : [OBDisparityParam](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all class members with links to the classes they belong to:

- ~ -

- ~AccelFrame() : [\*\*ob::AccelFrame\*\*](#)
- ~AccelStreamProfile() : [\*\*ob::AccelStreamProfile\*\*](#)
- ~Align() : [\*\*ob::Align\*\*](#)
- ~CameraParamList() : [\*\*ob::CameraParamList\*\*](#)
- ~ColorFrame() : [\*\*ob::ColorFrame\*\*](#)
- ~ConfidenceFrame() : [\*\*ob::ConfidenceFrame\*\*](#)
- ~Config() : [\*\*ob::Config\*\*](#)
- ~Context() : [\*\*ob::Context\*\*](#)
- ~DecimationFilter() : [\*\*ob::DecimationFilter\*\*](#)
- ~DepthFrame() : [\*\*ob::DepthFrame\*\*](#)
- ~Device() : [\*\*ob::Device\*\*](#)
- ~DeviceFrameInterleaveList() : [\*\*ob::DeviceFrameInterleaveList\*\*](#)
- ~DeviceInfo() : [\*\*ob::DeviceInfo\*\*](#)
- ~DeviceList() : [\*\*ob::DeviceList\*\*](#)
- ~DevicePresetList() : [\*\*ob::DevicePresetList\*\*](#)
- ~DisparityTransform() : [\*\*ob::DisparityTransform\*\*](#)
- ~Error() : [\*\*ob::Error\*\*](#)
- ~Filter() : [\*\*ob::Filter\*\*](#)
- ~FormatConvertFilter() : [\*\*ob::FormatConvertFilter\*\*](#)
- ~Frame() : [\*\*ob::Frame\*\*](#)
- ~FrameSet() : [\*\*ob::FrameSet\*\*](#)
- ~GyroFrame() : [\*\*ob::GyroFrame\*\*](#)
- ~GyroStreamProfile() : [\*\*ob::GyroStreamProfile\*\*](#)
- ~HdrMerge() : [\*\*ob::HdrMerge\*\*](#)
- ~HoleFillingFilter() : [\*\*ob::HoleFillingFilter\*\*](#)
- ~IRFrame() : [\*\*ob::IRFrame\*\*](#)
- ~NoiseRemovalFilter() : [\*\*ob::NoiseRemovalFilter\*\*](#)
- ~OBDepthWorkModeList() : [\*\*ob::OBDepthWorkModeList\*\*](#)
- ~OBFilterList() : [\*\*ob::OBFilterList\*\*](#)
- ~Pipeline() : [\*\*ob::Pipeline\*\*](#)
- ~PlaybackDevice() : [\*\*ob::PlaybackDevice\*\*](#)
- ~PointCloudFilter() : [\*\*ob::PointCloudFilter\*\*](#)
- ~PointsFrame() : [\*\*ob::PointsFrame\*\*](#)
- ~PresetResolutionConfigList() : [\*\*ob::PresetResolutionConfigList\*\*](#)
- ~RecordDevice() : [\*\*ob::RecordDevice\*\*](#)
- ~Sensor() : [\*\*ob::Sensor\*\*](#)
- ~SensorList() : [\*\*ob::SensorList\*\*](#)
- ~SequenceIdFilter() : [\*\*ob::SequenceIdFilter\*\*](#)
- ~SpatialAdvancedFilter() : [\*\*ob::SpatialAdvancedFilter\*\*](#)
- ~StreamProfile() : [\*\*ob::StreamProfile\*\*](#)
- ~StreamProfileList() : [\*\*ob::StreamProfileList\*\*](#)

- ~TemporalFilter() : **ob::TemporalFilter**
- ~ThresholdFilter() : **ob::ThresholdFilter**
- ~VideoFrame() : **ob::VideoFrame**
- ~VideoStreamProfile() : **ob::VideoStreamProfile**

Generated on for OrbbeeSDK by **doxygen** 1.14.0

Here is a list of all file members with links to the files they belong to:

- a -

- ADAPTIVE\_PERFORMANCE\_MODE : [ObTypes.h](#)
- ALIGN\_D2C\_HW\_MODE : [ObTypes.h](#)
- ALIGN\_D2C\_SW\_MODE : [ObTypes.h](#)
- ALIGN\_DISABLE : [ObTypes.h](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- d -

- DATA\_TRAN\_ERR\_BUSY : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_OTHER : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_TRAN\_FAILED : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_UNSUPPORTED : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_VERIFY\_FAILED : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_DONE : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_STOPPED : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_TRANSFERRING : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_VERIFY\_DONE : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_VERIFYING : [ObTypes.h](#)
- DEPTH\_CROPPING\_MODE\_AUTO : [ObTypes.h](#)
- DEPTH\_CROPPING\_MODE\_CLOSE : [ObTypes.h](#)
- DEPTH\_CROPPING\_MODE\_OPEN : [ObTypes.h](#)
- DEVICE\_IP\_ADDR\_CONFIG : [ObTypes.h](#)
- DEVICE\_LOG\_SEVERITY\_LEVEL : [ObTypes.h](#)
- DEVICE\_TEMPERATURE : [ObTypes.h](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all macros with links to the files they belong to:

- e -

- enableMultiDeviceSync : [Context.hpp](#)

- f -

- FORMAT\_I420\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_BGR888 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_BGRA : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_I420 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_NV21 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_BGR888 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_NV12\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_NV21\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_RGB888\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_UYVY\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_RGB888 : [ObTypes.h](#)

- g -

- getPositionValueScale : [Frame.hpp](#)

- i -

- IS\_FIXED\_SIZE\_FORMAT : [ObTypes.h](#)
- is\_ir\_frame : [ObTypes.h](#)
- is\_ir\_sensor : [ObTypes.h](#)
- is\_ir\_stream : [ObTypes.h](#)
- IS\_PACKED\_FORMAT : [ObTypes.h](#)
- isIRFrame : [ObTypes.h](#)
- isIRSensor : [ObTypes.h](#)
- isIRStream : [ObTypes.h](#)

- o -

- ob\_accel\_frame\_temperature : [Frame.h](#)
- ob\_accel\_frame\_value : [Frame.h](#)
- OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY : [ObTypes.h](#)
- OB\_ACCEL\_SAMPLE\_RATE\_ANY : [ObTypes.h](#)
- ob\_accel\_stream\_profile\_full\_scale\_range : [StreamProfile.h](#)
- ob\_accel\_stream\_profile\_sample\_rate : [StreamProfile.h](#)
- ob\_camera\_param\_list\_count : [Device.h](#)
- ob\_config\_set\_depth\_scale\_require : [Pipeline.h](#)

- OB\_DEFAULT\_DECRYPT\_KEY : [ObTypes.h](#)
- OB\_DEFAULT\_STRIDE\_BYTES : [ObTypes.h](#)
- OB\_DEPRECATED : [Export.h](#)
- OB\_DEPRECATED\_EXPORT : [Export.h](#)
- OB\_DEPRECATED\_NO\_EXPORT : [Export.h](#)
- ob\_depth\_work\_mode\_list\_count : [Advanced.h](#)
- ob\_device\_get\_supported\_property : [Device.h](#)
- ob\_device\_info\_asicName : [Device.h](#)
- ob\_device\_info\_connection\_type : [Device.h](#)
- ob\_device\_info\_device\_type : [Device.h](#)
- ob\_device\_info\_firmware\_version : [Device.h](#)
- ob\_device\_info\_hardware\_version : [Device.h](#)
- ob\_device\_info\_ip\_address : [Device.h](#)
- ob\_device\_info\_name : [Device.h](#)
- ob\_device\_info\_pid : [Device.h](#)
- ob\_device\_info\_serial\_number : [Device.h](#)
- ob\_device\_info\_supported\_min\_sdk\_version : [Device.h](#)
- ob\_device\_info\_uid : [Device.h](#)
- ob\_device\_info\_vid : [Device.h](#)
- ob\_device\_ip\_addr\_config : [ObTypes.h](#)
- ob\_device\_list\_device\_count : [Device.h](#)
- ob\_device\_list\_get\_device\_count : [Device.h](#)
- ob\_device\_list\_get\_extension\_info : [Device.h](#)
- ob\_device\_preset\_list\_count : [Advanced.h](#)
- ob\_device\_state\_changed : [Device.h](#)
- ob\_device\_timer\_reset : [MultipleDevices.h](#)
- ob\_device\_upgrade : [Device.h](#)
- ob\_device\_upgrade\_from\_data : [Device.h](#)
- ob\_enable\_multi\_device\_sync : [Context.h](#)
- ob\_error\_args : [Error.h](#)
- ob\_error\_exception\_type : [Error.h](#)
- ob\_error\_function : [Error.h](#)
- ob\_error\_message : [Error.h](#)
- ob\_error\_status : [Error.h](#)
- OB\_EXPORT : [Export.h](#)
- ob\_filter\_callback : [ObTypes.h](#)
- OB\_FORMAT\_ANY : [ObTypes.h](#)
- OB\_FORMAT\_MJPEG : [ObTypes.h](#)
- OB\_FORMAT\_RGB888 : [ObTypes.h](#)
- OB\_FPS\_ANY : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_FULL\_FRAME\_REQUIRE : [ObTypes.h](#)
- ob\_frame\_data : [Frame.h](#)
- ob\_frame\_data\_size : [Frame.h](#)
- ob\_frame\_format : [Frame.h](#)

- ob\_frame\_global\_time\_stamp\_us : **Frame.h**
- ob\_frame\_index : **Frame.h**
- ob\_frame\_metadata : **Frame.h**
- ob\_frame\_metadata\_size : **Frame.h**
- OB\_FRAME\_METADATA\_TYPE\_EMITTER\_MODE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER\_MODE : **ObTypes.h**
- ob\_frame\_set\_device\_time\_stamp : **Frame.h**
- ob\_frame\_set\_device\_time\_stamp\_us : **Frame.h**
- ob\_frame\_set\_system\_time\_stamp : **Frame.h**
- ob\_frame\_system\_time\_stamp : **Frame.h**
- ob\_frame\_system\_time\_stamp\_us : **Frame.h**
- ob\_frame\_time\_stamp : **Frame.h**
- ob\_frame\_time\_stamp\_us : **Frame.h**
- ob\_frameset\_color\_frame : **Frame.h**
- ob\_frameset\_depth\_frame : **Frame.h**
- ob\_frameset\_frame\_count : **Frame.h**
- ob\_frameset\_get\_frame\_count : **Frame.h**
- ob\_frameset\_ir\_frame : **Frame.h**
- ob\_frameset\_points\_frame : **Frame.h**
- ob\_get\_filter : **Filter.h**
- ob\_get\_filter\_name : **Filter.h**
- ob\_gyro\_frame\_temperature : **Frame.h**
- ob\_gyro\_frame\_value : **Frame.h**
- OB\_GYRO\_FULL\_SCALE\_RANGE\_ANY : **ObTypes.h**
- OB\_GYRO\_SAMPLE\_RATE\_ANY : **ObTypes.h**
- ob\_gyro\_stream\_profile\_full\_scale\_range : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_sample\_rate : **StreamProfile.h**
- OB\_HEIGHT\_ANY : **ObTypes.h**
- ob\_is\_video\_sensor\_type : **ObTypes.h**
- ob\_is\_video\_stream\_type : **ObTypes.h**
- OB\_LOG\_SEVERITY\_NONE : **ObTypes.h**
- OB\_NO\_EXPORT : **Export.h**
- OB\_PATH\_MAX : **ObTypes.h**
- ob\_playback\_callback : **ObTypes.h**
- ob\_points\_frame\_get\_position\_value\_scale : **Frame.h**
- OB\_PROFILE\_DEFAULT : **ObTypes.h**
- OB\_PROP\_DEPTH\_MAX\_DIFF\_INT : **Property.h**
- OB\_PROP\_DEPTH\_MAX\_SPECKLE\_SIZE\_INT : **Property.h**
- OB\_PROP\_DEPTH\_SOFT\_FILTER\_BOOL : **Property.h**
- OB\_PROP\_DEVICE\_USB3\_REPEAT\_IDENTIFY\_BOOL : **Property.h**
- OB\_PROP\_LASER\_ENERGY\_LEVEL\_INT : **Property.h**
- OB\_PROP\_LASER\_HW\_ENERGY\_LEVEL\_INT : **Property.h**
- OB\_PROP\_LASER\_ON\_OFF\_MODE\_INT : **Property.h**
- OB\_PROP\_TIMER\_RESET\_TRIGGER\_OUT\_ENABLE\_BOOL : **Property.h**

- ob\_sensor\_get\_recommended\_filter\_list : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor\_count : [Sensor.h](#)
- ob\_set\_logger\_callback : [Context.h](#)
- ob\_stream\_profile\_format : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_count : [StreamProfile.h](#)
- ob\_stream\_profile\_type : [StreamProfile.h](#)
- ob\_video\_frame\_height : [Frame.h](#)
- ob\_video\_frame\_metadata : [Frame.h](#)
- ob\_video\_frame\_metadata\_size : [Frame.h](#)
- ob\_video\_frame\_pixel\_available\_bit\_size : [Frame.h](#)
- ob\_video\_frame\_width : [Frame.h](#)
- ob\_video\_stream\_profile\_fps : [StreamProfile.h](#)
- ob\_video\_stream\_profile\_height : [StreamProfile.h](#)
- ob\_video\_stream\_profile\_width : [StreamProfile.h](#)
- OB\_WIDTH\_ANY : [ObTypes.h](#)
- OBDeviceIpAddrConfig : [ObTypes.h](#)

- t -

- timerReset : [Device.hpp](#)

Here is a list of all file members with links to the files they belong to:

- e -

- enableMultiDeviceSync : [Context.hpp](#)
- ERR\_DDR : [ObTypes.h](#)
- ERR\_ERASE : [ObTypes.h](#)
- ERR\_FLASH\_TYPE : [ObTypes.h](#)
- ERR\_IMAGE\_SIZE : [ObTypes.h](#)
- ERR\_INVALID\_COUNT : [ObTypes.h](#)
- ERR\_MISMATCH : [ObTypes.h](#)
- ERR\_OTHER : [ObTypes.h](#)
- ERR\_PROGRAM : [ObTypes.h](#)
- ERR\_TIMEOUT : [ObTypes.h](#)
- ERR\_UNSUPPORT\_DEV : [ObTypes.h](#)
- ERR\_VERIFY : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all enums with links to the files they belong to:

- 0 -

- OB\_CMD\_VERSION : [ObTypes.h](#)
- OB\_COORDINATE\_SYSTEM\_TYPE : [ObTypes.h](#)
- OB\_DDO\_NOISE\_REMOVAL\_TYPE : [ObTypes.h](#)
- OB\_DEVICE\_DEVELOPMENT\_MODE : [ObTypes.h](#)
- OB\_EDGE\_NOISE\_REMOVAL\_TYPE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_MODE : [ObTypes.h](#)
- ob\_frame\_metadata\_type : [ObTypes.h](#)
- ob\_multi\_device\_sync\_mode : [ObTypes.h](#)
- ob\_playback\_status : [ObTypes.h](#)
- ob\_power\_line\_freq\_mode : [ObTypes.h](#)
- ob\_rotate\_degree\_type : [ObTypes.h](#)
- ob\_uvc\_backend\_type : [ObTypes.h](#)
- OBAccelFullScaleRange : [ObTypes.h](#)
- OBAlignMode : [ObTypes.h](#)
- OBCameraDistortionModel : [ObTypes.h](#)
- OBCameraPerformanceMode : [ObTypes.h](#)
- OBCommunicationType : [ObTypes.h](#)
- OBCompressionMode : [ObTypes.h](#)
- OBConvertFormat : [ObTypes.h](#)
- OBDataTranState : [ObTypes.h](#)
- OBDCPowerState : [ObTypes.h](#)
- OBDepthCroppingMode : [ObTypes.h](#)
- OBDepthPrecisionLevel : [ObTypes.h](#)
- OBDepthWorkModeTag : [ObTypes.h](#)
- OBDeviceType : [ObTypes.h](#)
- OBExceptionType : [ObTypes.h](#)
- OBFileTranState : [ObTypes.h](#)
- OBFilterConfigValueType : [ObTypes.h](#)
- OBFormat : [ObTypes.h](#)
- OBFrameType : [ObTypes.h](#)
- OBGyroFullScaleRange : [ObTypes.h](#)
- OBHoleFillingMode : [ObTypes.h](#)
- OBIMUSampleRate : [ObTypes.h](#)
- OBLogSeverity : [ObTypes.h](#)
- OBMediaState : [ObTypes.h](#)
- OBMediaType : [ObTypes.h](#)
- OBPermissionType : [ObTypes.h](#)
- OBPixelType : [ObTypes.h](#)
- OBPropertyID : [Property.h](#)
- OB.PropertyType : [Property.h](#)
- OBSensorType : [ObTypes.h](#)

- OBStatus : [ObTypes.h](#)
- OBStreamType : [ObTypes.h](#)
- OBSyncMode : [ObTypes.h](#)
- OBTofFilterRange : [ObTypes.h](#)
- OBUpgradeState : [ObTypes.h](#)
- OBUSPowerState : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all enum values with links to the files they belong to:

- a -

- ADAPTIVE\_PERFORMANCE\_MODE : [ObTypes.h](#)
- ALIGN\_D2C\_HW\_MODE : [ObTypes.h](#)
- ALIGN\_D2C\_SW\_MODE : [ObTypes.h](#)
- ALIGN\_DISABLE : [ObTypes.h](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all enum values with links to the files they belong to:

- d -

- DATA\_TRAN\_ERR\_BUSY : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_OTHER : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_TRAN\_FAILED : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_UNSUPPORTED : [ObTypes.h](#)
- DATA\_TRAN\_ERR\_VERIFY\_FAILED : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_DONE : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_STOPPED : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_TRANSFERRING : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_VERIFY\_DONE : [ObTypes.h](#)
- DATA\_TRAN\_STAT\_VERIFYING : [ObTypes.h](#)
- DEPTH\_CROPPING\_MODE\_AUTO : [ObTypes.h](#)
- DEPTH\_CROPPING\_MODE\_CLOSE : [ObTypes.h](#)
- DEPTH\_CROPPING\_MODE\_OPEN : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all enum values with links to the files they belong to:

- e -

- ERR\_DDR : [ObTypes.h](#)
- ERR\_ERASE : [ObTypes.h](#)
- ERR\_FLASH\_TYPE : [ObTypes.h](#)
- ERR\_IMAGE\_SIZE : [ObTypes.h](#)
- ERR\_INVALID\_COUNT : [ObTypes.h](#)
- ERR\_MISMATCH : [ObTypes.h](#)
- ERR\_OTHER : [ObTypes.h](#)
- ERR\_PROGRAM : [ObTypes.h](#)
- ERR\_TIMEOUT : [ObTypes.h](#)
- ERR\_UN SUPPORT\_DEV : [ObTypes.h](#)
- ERR\_VERIFY : [ObTypes.h](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all enum values with links to the files they belong to:

- f -

- FILE\_TRAN\_ERR\_DDR : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_MD5\_ERROR : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_NOT\_ENOUGH\_SPACE : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_PATH\_NOT\_WRITABLE : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_TIMEOUT : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_WRITE\_FLASH\_ERROR : [ObTypes.h](#)
- FILE\_TRAN\_STAT\_DONE : [ObTypes.h](#)
- FILE\_TRAN\_STAT\_PREPAR : [ObTypes.h](#)
- FILE\_TRAN\_STAT\_TRANSFER : [ObTypes.h](#)
- FORMAT\_BGR\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_BGRA\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_I420\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_BGRA : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_I420 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_NV12 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_NV21 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_NV12\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_NV21\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_RGB\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_RGBA\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_UYVY\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_Y16\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_Y8\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_BGRA : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_RGBA : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_Y16 : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_Y8 : [ObTypes.h](#)

Here is a list of all enum values with links to the files they belong to:

- h -

- HIGH\_PERFORMANCE\_MODE : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all enum values with links to the files they belong to:

- 0 -

- OB\_ACCEL\_FS\_12g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_16g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_24g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_2g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_3g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_4g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_6g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_8g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_UNKNOWN : [ObTypes.h](#)
- OB\_BOOL\_PROPERTY : [Property.h](#)
- OB\_CMD\_VERSION\_INVALID : [ObTypes.h](#)
- OB\_CMD\_VERSION\_NOVERSION : [ObTypes.h](#)
- OB\_CMD\_VERSION\_V0 : [ObTypes.h](#)
- OB\_CMD\_VERSION\_V1 : [ObTypes.h](#)
- OB\_CMD\_VERSION\_V2 : [ObTypes.h](#)
- OB\_CMD\_VERSION\_V3 : [ObTypes.h](#)
- OB\_COMM\_NET : [ObTypes.h](#)
- OB\_COMM\_USB : [ObTypes.h](#)
- OB\_COMPRESSION\_LOSSLESS : [ObTypes.h](#)
- OB\_COMPRESSION\_LOSSY : [ObTypes.h](#)
- OB\_CUSTOM\_DEPTH\_WORK\_MODE : [ObTypes.h](#)
- OB\_DC\_POWER\_NO\_PLUGIN : [ObTypes.h](#)
- OB\_DC\_POWER\_PLUGIN : [ObTypes.h](#)
- OB\_DEVELOPER\_MODE : [ObTypes.h](#)
- OB\_DEVICE\_AUTO\_CAPTURE\_ENABLE\_BOOL : [Property.h](#)
- OB\_DEVICE\_AUTO\_CAPTURE\_INTERVAL\_TIME\_INT : [Property.h](#)
- OB\_DEVICE\_DEPTH\_WORK\_MODE : [ObTypes.h](#)
- OB\_DEVICE\_PTP\_CLOCK\_SYNC\_ENABLE\_BOOL : [Property.h](#)
- OB\_DEVICE\_TYPE\_UNKNOWN : [ObTypes.h](#)
- OB\_DISTORTION\_BROWN\_CONRADY : [ObTypes.h](#)
- OB\_DISTORTION\_BROWN\_CONRADY\_K6 : [ObTypes.h](#)
- OB\_DISTORTION\_INVERSE\_BROWN\_CONRADY : [ObTypes.h](#)
- OB\_DISTORTION\_KANNALA\_BRANDT4 : [ObTypes.h](#)
- OB\_DISTORTION\_MODIFIED\_BROWN\_CONRADY : [ObTypes.h](#)
- OB\_DISTORTION\_NONE : [ObTypes.h](#)
- OB\_EXCEPTION\_STD\_EXCEPTION : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_CAMERA\_DISCONNECTED : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_INVALID\_VALUE : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_IO : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_MEMORY : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_NOT\_IMPLEMENTED : [ObTypes.h](#)

- OB\_EXCEPTION\_TYPE\_PLATFORM : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_UNKNOWN : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_UNSUPPORTED\_OPERATION : [ObTypes.h](#)
- OB\_EXCEPTION\_TYPE\_WRONG\_API\_CALL\_SEQUENCE : [ObTypes.h](#)
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_BOOLEAN : [ObTypes.h](#)
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_FLOAT : [ObTypes.h](#)
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_INT : [ObTypes.h](#)
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_INVALID : [ObTypes.h](#)
- OB\_FLOAT\_PROPERTY : [Property.h](#)
- OB\_FORMAT\_ACCEL : [ObTypes.h](#)
- OB\_FORMAT\_BA81 : [ObTypes.h](#)
- OB\_FORMAT\_BGR : [ObTypes.h](#)
- OB\_FORMAT\_BGRA : [ObTypes.h](#)
- OB\_FORMAT\_BYR2 : [ObTypes.h](#)
- OB\_FORMAT\_COMPRESSED : [ObTypes.h](#)
- OB\_FORMAT\_GRAY : [ObTypes.h](#)
- OB\_FORMAT\_GYRO : [ObTypes.h](#)
- OB\_FORMAT\_H264 : [ObTypes.h](#)
- OB\_FORMAT\_H265 : [ObTypes.h](#)
- OB\_FORMAT\_HEVC : [ObTypes.h](#)
- OB\_FORMAT\_I420 : [ObTypes.h](#)
- OB\_FORMAT\_MJPG : [ObTypes.h](#)
- OB\_FORMAT\_NV12 : [ObTypes.h](#)
- OB\_FORMAT\_NV21 : [ObTypes.h](#)
- OB\_FORMAT\_POINT : [ObTypes.h](#)
- OB\_FORMAT\_RGB : [ObTypes.h](#)
- OB\_FORMAT\_RGB\_POINT : [ObTypes.h](#)
- OB\_FORMAT\_RGBA : [ObTypes.h](#)
- OB\_FORMAT\_RLE : [ObTypes.h](#)
- OB\_FORMAT\_RVL : [ObTypes.h](#)
- OB\_FORMAT\_RW16 : [ObTypes.h](#)
- OB\_FORMAT\_UNKNOWN : [ObTypes.h](#)
- OB\_FORMAT\_UYVY : [ObTypes.h](#)
- OB\_FORMAT\_Y10 : [ObTypes.h](#)
- OB\_FORMAT\_Y11 : [ObTypes.h](#)
- OB\_FORMAT\_Y12 : [ObTypes.h](#)
- OB\_FORMAT\_Y12C4 : [ObTypes.h](#)
- OB\_FORMAT\_Y14 : [ObTypes.h](#)
- OB\_FORMAT\_Y16 : [ObTypes.h](#)
- OB\_FORMAT\_Y8 : [ObTypes.h](#)
- OB\_FORMAT\_YUY2 : [ObTypes.h](#)
- OB\_FORMAT\_YUYV : [ObTypes.h](#)
- OB\_FORMAT\_YV12 : [ObTypes.h](#)
- OB\_FORMAT\_Z16 : [ObTypes.h](#)

- OB\_FRAME\_ACCEL : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_ALL\_TYPE\_FRAME\_REQUIRE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_ANY\_SITUATION : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_COLOR\_FRAME\_REQUIRE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_DISABLE : [ObTypes.h](#)
- OB\_FRAME\_COLOR : [ObTypes.h](#)
- OB\_FRAME\_CONFIDENCE : [ObTypes.h](#)
- OB\_FRAME\_DEPTH : [ObTypes.h](#)
- OB\_FRAME\_GYRO : [ObTypes.h](#)
- OB\_FRAME\_IR : [ObTypes.h](#)
- OB\_FRAME\_IR\_LEFT : [ObTypes.h](#)
- OB\_FRAME\_IR\_RIGHT : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_ACTUAL\_FRAME\_RATE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_BOTTOM : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_LEFT : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_RIGHT : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_TOP : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_AUTO\_EXPOSURE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_AUTO\_WHITE\_BALANCE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_BACKLIGHT\_COMPENSATION : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_BRIGHTNESS : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_CONTRAST : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_COUNT : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_DISPARITY\_SEARCH\_OFFSET : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_DISPARITY\_SEARCH\_RANGE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_EXPOSURE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_EXPOSURE\_PRIORITY : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_FRAME\_NUMBER : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_FRAME\_RATE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_GAIN : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_GAMMA : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_GPIO\_INPUT\_DATA : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_HDR\_SEQUENCE\_INDEX : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_HDR\_SEQUENCE\_NAME : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_HDR\_SEQUENCE\_SIZE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_HUE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER\_LEVEL : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_LASER\_STATUS : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_LOW\_LIGHT\_COMPENSATION : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_MANUAL\_WHITE\_BALANCE : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_POWER\_LINE\_FREQUENCY : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_SATURATION : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_SENSOR\_TIMESTAMP : [ObTypes.h](#)

- OB\_FRAME\_METADATA\_TYPE\_SHARPNESS : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_TIMESTAMP : [ObTypes.h](#)
- OB\_FRAME\_METADATA\_TYPE\_WHITE\_BALANCE : [ObTypes.h](#)
- OB\_FRAME\_POINTS : [ObTypes.h](#)
- OB\_FRAME\_RAW\_PHASE : [ObTypes.h](#)
- OB\_FRAME\_SET : [ObTypes.h](#)
- OB\_FRAME\_TYPE\_COUNT : [ObTypes.h](#)
- OB\_FRAME\_UNKNOWN : [ObTypes.h](#)
- OB\_FRAME\_VIDEO : [ObTypes.h](#)
- OB\_GYRO\_FS\_1000dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_125dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_16dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_2000dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_250dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_31dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_400dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_500dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_62dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_800dps : [ObTypes.h](#)
- OB\_GYRO\_FS\_UNKNOWN : [ObTypes.h](#)
- OB\_HOLE\_FILL\_FAREST : [ObTypes.h](#)
- OB\_HOLE\_FILL\_NEAREST : [ObTypes.h](#)
- OB\_HOLE\_FILL\_TOP : [ObTypes.h](#)
- OB\_INT\_PROPERTY : [Property.h](#)
- OB\_LEFT\_HAND\_COORDINATE\_SYSTEM : [ObTypes.h](#)
- OB\_LOG\_SEVERITY\_DEBUG : [ObTypes.h](#)
- OB\_LOG\_SEVERITY\_ERROR : [ObTypes.h](#)
- OB\_LOG\_SEVERITY\_FATAL : [ObTypes.h](#)
- OB\_LOG\_SEVERITY\_INFO : [ObTypes.h](#)
- OB\_LOG\_SEVERITY\_OFF : [ObTypes.h](#)
- OB\_LOG\_SEVERITY\_WARN : [ObTypes.h](#)
- OB\_MEDIA\_ACCEL\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_ALL : [ObTypes.h](#)
- OB\_MEDIA\_BEGIN : [ObTypes.h](#)
- OB\_MEDIA\_CAMERA\_PARAM : [ObTypes.h](#)
- OB\_MEDIA\_COLOR\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_DEPTH\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_DEVICE\_INFO : [ObTypes.h](#)
- OB\_MEDIA\_END : [ObTypes.h](#)
- OB\_MEDIA\_GYRO\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_IR\_LEFT\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_IR\_RIGHT\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_IR\_STREAM : [ObTypes.h](#)
- OB\_MEDIA\_PAUSE : [ObTypes.h](#)

- OB\_MEDIA\_RESUME : **ObTypes.h**
- OB\_MEDIA\_STREAM\_INFO : **ObTypes.h**
- OB\_MG\_FILTER : **ObTypes.h**
- OB\_MGA\_FILTER : **ObTypes.h**
- OB\_MGC\_FILTER : **ObTypes.h**
- OB\_MGH\_FILTER : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_FREE\_RUN : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_HARDWARE\_TRIGGERING : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_IR\_IMU\_SYNC : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_PRIMARY : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_SECONDARY : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_SECONDARY\_SYNCED : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_SOFTWARE\_TRIGGERING : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_STANDALONE : **ObTypes.h**
- OB\_NR\_LUT : **ObTypes.h**
- OB\_NR\_OVERALL : **ObTypes.h**
- OB\_PERMISSION\_ANY : **ObTypes.h**
- OB\_PERMISSION\_DENY : **ObTypes.h**
- OB\_PERMISSION\_READ : **ObTypes.h**
- OB\_PERMISSION\_READ\_WRITE : **ObTypes.h**
- OB\_PERMISSION\_WRITE : **ObTypes.h**
- OB\_PIXEL\_DEPTH : **ObTypes.h**
- OB\_PIXEL\_DISPARITY : **ObTypes.h**
- OB\_PIXEL\_RAW\_PHASE : **ObTypes.h**
- OB\_PIXEL\_TOF\_DEPTH : **ObTypes.h**
- OB\_PIXEL\_UNKNOWN : **ObTypes.h**
- OB\_PLAYBACK\_COUNT : **ObTypes.h**
- OB\_PLAYBACK\_PAUSED : **ObTypes.h**
- OB\_PLAYBACK\_PLAYING : **ObTypes.h**
- OB\_PLAYBACK\_STOPPED : **ObTypes.h**
- OB\_PLAYBACK\_UNKNOWN : **ObTypes.h**
- OB\_POWER\_LINE\_FREQ\_MODE\_50HZ : **ObTypes.h**
- OB\_POWER\_LINE\_FREQ\_MODE\_60HZ : **ObTypes.h**
- OB\_POWER\_LINE\_FREQ\_MODE\_CLOSE : **ObTypes.h**
- OB\_PRECISION\_0MM05 : **ObTypes.h**
- OB\_PRECISION\_0MM1 : **ObTypes.h**
- OB\_PRECISION\_0MM2 : **ObTypes.h**
- OB\_PRECISION\_0MM4 : **ObTypes.h**
- OB\_PRECISION\_0MM5 : **ObTypes.h**
- OB\_PRECISION\_0MM8 : **ObTypes.h**
- OB\_PRECISION\_1MM : **ObTypes.h**
- OB\_PRECISION\_COUNT : **ObTypes.h**
- OB\_PRECISION\_UNKNOWN : **ObTypes.h**
- OB\_PROP\_ANTI\_COLLUSION\_ACTIVATION\_STATUS\_BOOL : **Property.h**

- OB\_PROP\_BOOT\_INTO\_RECOVERY\_MODE\_BOOL : [Property.h](#)
- OB\_PROP\_BRT\_BOOL : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_FRAME\_NUMBER\_INT : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_NUMBER\_INTERVAL\_INT : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_SIGNAL\_BOOL : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_TIME\_INTERVAL\_INT : [Property.h](#)
- OB\_PROP\_CAPTURE\_INTERVAL\_MODE\_INT : [Property.h](#)
- OB\_PROP\_CHECK\_PPS\_SYNC\_IN\_SIGNAL\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_AE\_MAX\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_AUTO\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_AUTO\_EXPOSURE\_PRIORITY\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_AUTO\_WHITE\_BALANCE\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_BACKLIGHT\_COMPENSATION\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_BRIGHTNESS\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_CONTRAST\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_FOCUS\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_GAMMA\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_HDR\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_HUE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_MAXIMAL\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_MAXIMAL\_SHUTTER\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_POWER\_LINE\_FREQUENCY\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_ROLL\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_SATURATION\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_SHARPNESS\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_SHUTTER\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_WHITE\_BALANCE\_INT : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_THRESHOLD\_INT : [Property.h](#)
- OB\_PROP\_D2C\_PREPROCESS\_BOOL : [Property.h](#)
- OB\_PROP\_DC\_POWER\_STATE\_INT : [Property.h](#)
- OB\_PROP\_DEBUG\_ESGM\_CONFIDENCE\_FLOAT : [Property.h](#)
- OB\_PROP\_DEPTH\_ALIGN\_HARDWARE\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_ALIGN\_HARDWARE\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_AUTO\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_AUTO\_EXPOSURE\_PRIORITY\_INT : [Property.h](#)

- OB\_PROP\_DEPTH\_CROPPING\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_HOLEFILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_MAX\_DIFF\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_MAX\_SPECKLE\_SIZE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_POSTFILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_PRECISION\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_RM\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_UNIT\_FLEXIBLE\_ADJUSTMENT\_FLOAT : [Property.h](#)
- OB\_PROP\_DEPTH\_WITH\_CONFIDENCE\_STREAM\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_COMMUNICATION\_TYPE\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_DEVELOPMENT\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_IN\_RECOVERY\_MODE\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_PERFORMANCE\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_REBOOT\_DELAY\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_REPOWER\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_USB2\_REPEAT\_IDENTIFY\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_WORK\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DISP\_SEARCH\_OFFSET\_INT : [Property.h](#)
- OB\_PROP\_DISP\_SEARCH\_RANGE\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DISPARITY\_TO\_DEPTH\_BOOL : [Property.h](#)
- OB\_PROP\_EXTERNAL\_SIGNAL\_RESET\_BOOL : [Property.h](#)
- OB\_PROP\_FAN\_WORK\_MODE\_INT : [Property.h](#)
- OB\_PROP\_FLOOD\_BOOL : [Property.h](#)
- OB\_PROP\_FLOOD\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_FRAME\_INTERLEAVE\_CONFIG\_INDEX\_INT : [Property.h](#)
- OB\_PROP\_FRAME\_INTERLEAVE\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_FRAME\_INTERLEAVE\_LASER\_PATTERN\_SYNC\_DELAY\_INT : [Property.h](#)
- OB\_PROP\_GPM\_BOOL : [Property.h](#)
- OB\_PROP\_HARDWARE\_DISTORTION\_SWITCH\_BOOL : [Property.h](#)
- OB\_PROP\_HDR\_MERGE\_BOOL : [Property.h](#)
- OB\_PROP\_HEARTBEAT\_BOOL : [Property.h](#)
- OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_THRESHOLD\_FLOAT : [Property.h](#)
- OB\_PROP\_INDICATOR\_LIGHT\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_AE\_MAX\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_IR\_AUTO\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_BRIGHTNESS\_INT : [Property.h](#)
- OB\_PROP\_IR\_CHANNEL\_DATA\_SOURCE\_INT : [Property.h](#)

- OB\_PROP\_IR\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_IR\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_IR\_LONG\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RECTIFY\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RIGHT\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RIGHT\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RIGHT\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_IR\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_IR\_SHORT\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_ALWAYS\_ON\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_CONTROL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_CURRENT\_FLOAT : [Property.h](#)
- OB\_PROP\_LASER\_HIGH\_TEMPERATURE\_PROTECT\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_MODE\_INT : [Property.h](#)
- OB\_PROP\_LASER\_ON\_OFF\_PATTERN\_INT : [Property.h](#)
- OB\_PROP\_LASER\_OVERCURRENT\_PROTECTION\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_POWER\_ACTUAL\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_POWER\_LEVEL\_CONTROL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_PULSE\_WIDTH\_INT : [Property.h](#)
- OB\_PROP\_LASER\_PULSE\_WIDTH\_PROTECTION\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_LDP\_BOOL : [Property.h](#)
- OB\_PROP\_LDP\_MEASURE\_DISTANCE\_INT : [Property.h](#)
- OB\_PROP\_LDP\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_LOW\_EXPOSURE\_LASER\_CONTROL\_BOOL : [Property.h](#)
- OB\_PROP\_MAX\_DEPTH\_INT : [Property.h](#)
- OB\_PROP\_MIN\_DEPTH\_INT : [Property.h](#)
- OB\_PROP\_NETWORK\_BANDWIDTH\_TYPE\_INT : [Property.h](#)
- OB\_PROP\_ON\_CHIP\_CALIBRATION\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_ON\_CHIP\_CALIBRATION\_HEALTH\_CHECK\_FLOAT : [Property.h](#)
- OB\_PROP\_RECTIFY2\_BOOL : [Property.h](#)
- OB\_PROP\_RESTORE\_FACTORY\_SETTINGS\_BOOL : [Property.h](#)
- OB\_PROP\_RGB\_CUSTOM\_CROP\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_ACCEL\_FRAME\_TRANSFORMED\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_DEPTH\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_DISPARITY\_TO\_DEPTH\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_GYRO\_FRAME\_TRANSFORMED\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_IR\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_IR\_LEFT\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_IR\_RIGHT\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SKIP\_FRAME\_BOOL : [Property.h](#)
- OB\_PROP\_SLAVE\_DEVICE\_SYNC\_STATUS\_BOOL : [Property.h](#)

- OB\_PROP\_SWITCH\_IR\_MODE\_INT : [Property.h](#)
- OB\_PROP\_SYNC\_SIGNAL\_TRIGGER\_OUT\_BOOL : [Property.h](#)
- OB\_PROP\_TEMPERATURE\_COMPENSATION\_BOOL : [Property.h](#)
- OB\_PROP\_TIMER\_RESET\_DELAY\_US\_INT : [Property.h](#)
- OB\_PROP\_TIMER\_RESET\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_TIMER\_RESET\_SIGNAL\_BOOL : [Property.h](#)
- OB\_PROP\_TIMER\_RESET\_TRIGGER\_OUT\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_TIMESTAMP\_OFFSET\_INT : [Property.h](#)
- OB\_PROP\_TOF\_FILTER\_RANGE\_INT : [Property.h](#)
- OB\_PROP\_USB\_POWER\_STATE\_INT : [Property.h](#)
- OB\_PROP\_WATCHDOG\_BOOL : [Property.h](#)
- OB\_RAW\_DATA\_CAMERA\_CALIB\_JSON\_FILE : [Property.h](#)
- OB\_RIGHT\_HAND\_COORDINATE\_SYSTEM : [ObTypes.h](#)
- OB\_ROTATE\_DEGREE\_0 : [ObTypes.h](#)
- OB\_ROTATE\_DEGREE\_180 : [ObTypes.h](#)
- OB\_ROTATE\_DEGREE\_270 : [ObTypes.h](#)
- OB\_ROTATE\_DEGREE\_90 : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_100\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_12\_5\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_16\_KHZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_1\_5625\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_1\_KHZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_200\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_25\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_2\_KHZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_32\_KHZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_3\_125\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_400\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_4\_KHZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_500\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_50\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_6\_25\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_800\_HZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_8\_KHZ : [ObTypes.h](#)
- OB\_SAMPLE\_RATE\_UNKNOWN : [ObTypes.h](#)
- OB\_SENSOR\_ACCEL : [ObTypes.h](#)
- OB\_SENSOR\_COLOR : [ObTypes.h](#)
- OB\_SENSOR\_CONFIDENCE : [ObTypes.h](#)
- OB\_SENSOR\_DEPTH : [ObTypes.h](#)
- OB\_SENSOR\_GYRO : [ObTypes.h](#)
- OB\_SENSOR\_IR : [ObTypes.h](#)
- OB\_SENSOR\_IR\_LEFT : [ObTypes.h](#)
- OB\_SENSOR\_IR\_RIGHT : [ObTypes.h](#)
- OB\_SENSOR\_RAW\_PHASE : [ObTypes.h](#)

- OB\_SENSOR\_TYPE\_COUNT : [ObTypes.h](#)
- OB\_SENSOR\_UNKNOWN : [ObTypes.h](#)
- OB\_STATUS\_ERROR : [ObTypes.h](#)
- OB\_STATUS\_OK : [ObTypes.h](#)
- OB\_STREAM\_ACCEL : [ObTypes.h](#)
- OB\_STREAM\_COLOR : [ObTypes.h](#)
- OB\_STREAM\_CONFIDENCE : [ObTypes.h](#)
- OB\_STREAM\_DEPTH : [ObTypes.h](#)
- OB\_STREAM\_GYRO : [ObTypes.h](#)
- OB\_STREAM\_IR : [ObTypes.h](#)
- OB\_STREAM\_IR\_LEFT : [ObTypes.h](#)
- OB\_STREAM\_IR\_RIGHT : [ObTypes.h](#)
- OB\_STREAM\_RAW\_PHASE : [ObTypes.h](#)
- OB\_STREAM\_TYPE\_COUNT : [ObTypes.h](#)
- OB\_STREAM\_UNKNOWN : [ObTypes.h](#)
- OB\_STREAM\_VIDEO : [ObTypes.h](#)
- OB\_STRUCT ASIC\_SERIAL\_NUMBER : [Property.h](#)
- OB\_STRUCT\_BASELINE\_CALIBRATION\_PARAM : [Property.h](#)
- OB\_STRUCT\_COLOR\_AE\_ROI : [Property.h](#)
- OB\_STRUCT\_CURRENT\_DEPTH\_ALG\_MODE : [Property.h](#)
- OB\_STRUCT\_DEPTH\_AE\_ROI : [Property.h](#)
- OB\_STRUCT\_DEPTH\_HDR\_CONFIG : [Property.h](#)
- OB\_STRUCT\_DEPTH\_PRECISION\_SUPPORT\_LIST : [Property.h](#)
- OB\_STRUCT\_DEVICE\_IP\_ADDR\_CONFIG : [Property.h](#)
- OB\_STRUCT\_DEVICE\_SERIAL\_NUMBER : [Property.h](#)
- OB\_STRUCT\_DEVICE\_STATIC\_IP\_CONFIG\_RECORD : [Property.h](#)
- OB\_STRUCT\_DEVICE\_TEMPERATURE : [Property.h](#)
- OB\_STRUCT\_DEVICE\_TIME : [Property.h](#)
- OB\_STRUCT\_DISP\_OFFSET\_CONFIG : [Property.h](#)
- OB\_STRUCT\_MULTI\_DEVICE\_SYNC\_CONFIG : [Property.h](#)
- OB\_STRUCT\_PRESET\_RESOLUTION\_CONFIG : [Property.h](#)
- OB\_STRUCT\_PROPERTY : [Property.h](#)
- OB\_STRUCT\_RGB\_CROP\_ROI : [Property.h](#)
- OB\_STRUCT\_TOF\_EXPOSURE\_THRESHOLD\_CONTROL : [Property.h](#)
- OB\_STRUCTURED\_LIGHT\_BINOCULAR\_CAMERA : [ObTypes.h](#)
- OB\_STRUCTURED\_LIGHT\_MONOCULAR\_CAMERA : [ObTypes.h](#)
- OB\_SYNC\_MODE\_CLOSE : [ObTypes.h](#)
- OB\_SYNC\_MODE\_IR\_IMU\_SYNC : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY\_IR\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY MCU\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY\_SOFT\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_SECONDARY : [ObTypes.h](#)
- OB\_SYNC\_MODE\_SECONDARY\_SOFT\_TRIGGER : [ObTypes.h](#)

- OB\_SYNC\_MODE\_STANDALONE : [ObTypes.h](#)
- OB\_SYNC\_MODE\_UNKNOWN : [ObTypes.h](#)
- OB\_TOF\_CAMERA : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_CLOSE : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_DEBUG : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_LONG : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_MIDDLE : [ObTypes.h](#)
- OB\_USB\_POWER\_5V\_0A9 : [ObTypes.h](#)
- OB\_USB\_POWER\_5V\_1A5 : [ObTypes.h](#)
- OB\_USB\_POWER\_5V\_3A0 : [ObTypes.h](#)
- OB\_USB\_POWER\_NO\_PLUGIN : [ObTypes.h](#)
- OB\_USER\_MODE : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_AUTO : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_LIBUVC : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_MSMF : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_V4L2 : [ObTypes.h](#)

Here is a list of all enum values with links to the files they belong to:

- S -

- STAT\_DONE : [ObTypes.h](#)
- STAT\_DONE\_WITH\_DUPLICATES : [ObTypes.h](#)
- STAT\_FILE\_TRANSFER : [ObTypes.h](#)
- STAT\_IN\_PROGRESS : [ObTypes.h](#)
- STAT\_START : [ObTypes.h](#)
- STAT\_VERIFY\_IMAGE : [ObTypes.h](#)
- STAT\_VERIFY\_SUCCESS : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- f -

- FILE\_TRAN\_ERR\_DDR : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_MD5\_ERROR : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_NOT\_ENOUGH\_SPACE : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_PATH\_NOT\_WRITABLE : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_TIMEOUT : [ObTypes.h](#)
- FILE\_TRAN\_ERR\_WRITE\_FLASH\_ERROR : [ObTypes.h](#)
- FILE\_TRAN\_STAT\_DONE : [ObTypes.h](#)
- FILE\_TRAN\_STAT\_PREPAR : [ObTypes.h](#)
- FILE\_TRAN\_STAT\_TRANSFER : [ObTypes.h](#)
- FORMAT\_BGR\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_BGRA\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_I420\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_I420\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_BGR888 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_BGRA : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_I420 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_NV21 : [ObTypes.h](#)
- FORMAT\_MJPEG\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_BGR888 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_BGRA : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_I420 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_NV12 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_NV21 : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_MJPG\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_NV12\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_NV12\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_NV21\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_NV21\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_RGB888\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_RGB\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_RGBA\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_UYVY\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_UYVY\_TO\_RGB888 : [ObTypes.h](#)
- FORMAT\_Y16\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_Y8\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_BGR : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_BGRA : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_RGB : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_RGB888 : [ObTypes.h](#)

- FORMAT\_YUYV\_TO\_RGBA : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_Y16 : [ObTypes.h](#)
- FORMAT\_YUYV\_TO\_Y8 : [ObTypes.h](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the files they belong to:

- 0 -

- `ob_accel_frame_get_temperature()` : [Frame.h](#)
- `ob_accel_frame_get_value()` : [Frame.h](#)
- `ob_accel_range_type_to_string()` : [TypeHelper.h](#)
- `ob_accel_stream_profile_get_full_scale_range()` : [StreamProfile.h](#)
- `ob_accel_stream_profile_get_intrinsic()` : [StreamProfile.h](#)
- `ob_accel_stream_profile_get_sample_rate()` : [StreamProfile.h](#)
- `ob_accel_stream_profile_set_intrinsic()` : [StreamProfile.h](#)
- `ob_align_filter_set_align_to_stream_profile()` : [Filter.h](#)
- `ob_calibration_2d_to_2d()` : [Utils.h](#)
- `ob_calibration_2d_to_3d()` : [Utils.h](#)
- `ob_calibration_3d_to_2d()` : [Utils.h](#)
- `ob_calibration_3d_to_3d()` : [Utils.h](#)
- `ob_camera_param_list_get_count()` : [Device.h](#)
- `ob_camera_param_list_get_param()` : [Device.h](#)
- `ob_config_disable_all_stream()` : [Pipeline.h](#)
- `ob_config_disable_stream()` : [Pipeline.h](#)
- `ob_config_enable_accel_stream()` : [Pipeline.h](#)
- `ob_config_enable_all_stream()` : [Pipeline.h](#)
- `ob_config_enable_gyro_stream()` : [Pipeline.h](#)
- `ob_config_enable_stream()` : [Pipeline.h](#)
- `ob_config_enable_stream_with_stream_profile()` : [Pipeline.h](#)
- `ob_config_enable_video_stream()` : [Pipeline.h](#)
- `ob_config_get_enabled_stream_profile_list()` : [Pipeline.h](#)
- `ob_config_set_align_mode()` : [Pipeline.h](#)
- `ob_config_set_depth_scale_after_align_require()` : [Pipeline.h](#)
- `ob_config_set_frame_aggregate_output_mode()` : [Pipeline.h](#)
- `ob_create_accel_stream_profile()` : [StreamProfile.h](#)
- `ob_create_config()` : [Pipeline.h](#)
- `ob_create_context()` : [Context.h](#)
- `ob_create_context_with_config()` : [Context.h](#)
- `ob_create_error()` : [Error.h](#)
- `ob_create_filter()` : [Filter.h](#)
- `ob_create_frame()` : [Frame.h](#)
- `ob_create_frame_from_buffer()` : [Frame.h](#)
- `ob_create_frame_from_other_frame()` : [Frame.h](#)
- `ob_create_frame_from_stream_profile()` : [Frame.h](#)
- `ob_create_frameset()` : [Frame.h](#)
- `ob_create_gyro_stream_profile()` : [StreamProfile.h](#)
- `ob_create_net_device()` : [Context.h](#)
- `ob_create_pipeline()` : [Pipeline.h](#)
- `ob_create_pipeline_with_device()` : [Pipeline.h](#)

- ob\_create\_playback\_device() : **RecordPlayback.h**
- ob\_create\_private\_filter() : **Filter.h**
- ob\_create\_record\_device() : **RecordPlayback.h**
- ob\_create\_stream\_profile() : **StreamProfile.h**
- ob\_create\_stream\_profile\_from\_other\_stream\_profile() : **StreamProfile.h**
- ob\_create\_stream\_profile\_with\_new\_format() : **StreamProfile.h**
- ob\_create\_video\_frame() : **Frame.h**
- ob\_create\_video\_frame\_from\_buffer() : **Frame.h**
- ob\_create\_video\_stream\_profile() : **StreamProfile.h**
- ob\_delete\_camera\_param\_list() : **Device.h**
- ob\_delete\_config() : **Pipeline.h**
- ob\_delete\_context() : **Context.h**
- ob\_delete\_depth\_work\_mode\_list() : **Advanced.h**
- ob\_delete\_device() : **Device.h**
- ob\_delete\_device\_info() : **Device.h**
- ob\_delete\_device\_list() : **Device.h**
- ob\_delete\_error() : **Error.h**
- ob\_delete\_filter() : **Filter.h**
- ob\_delete\_filter\_config\_schema\_list() : **Filter.h**
- ob\_delete\_filter\_list() : **Filter.h**
- ob\_delete\_frame() : **Frame.h**
- ob\_delete\_frame\_interleave\_list() : **Advanced.h**
- ob\_delete\_pipeline() : **Pipeline.h**
- ob\_delete\_preset\_list() : **Advanced.h**
- ob\_delete\_preset\_resolution\_config\_list() : **Advanced.h**
- ob\_delete\_record\_device() : **RecordPlayback.h**
- ob\_delete\_sensor() : **Sensor.h**
- ob\_delete\_sensor\_list() : **Sensor.h**
- ob\_delete\_stream\_profile() : **StreamProfile.h**
- ob\_delete\_stream\_profile\_list() : **StreamProfile.h**
- ob\_depth\_frame\_get\_value\_scale() : **Frame.h**
- ob\_depth\_frame\_set\_value\_scale() : **Frame.h**
- ob\_depth\_work\_mode\_list\_get\_count() : **Advanced.h**
- ob\_depth\_work\_mode\_list\_get\_item() : **Advanced.h**
- ob\_device\_enable\_global\_timestamp() : **Device.h**
- ob\_device\_enable\_heartbeat() : **Device.h**
- ob\_device\_export\_current\_settings\_as\_preset\_json\_data() : **Advanced.h**
- ob\_device\_export\_current\_settings\_as\_preset\_json\_file() : **Advanced.h**
- ob\_device\_frame\_interleave\_list\_get\_count() : **Advanced.h**
- ob\_device\_frame\_interleave\_list\_get\_name() : **Advanced.h**
- ob\_device\_frame\_interleave\_list\_has\_frame\_interleave() : **Advanced.h**
- ob\_device\_get\_available\_frame\_interleave\_list() : **Advanced.h**
- ob\_device\_get\_available\_preset\_list() : **Advanced.h**
- ob\_device\_get\_available\_preset\_resolution\_config\_list() : **Advanced.h**

- ob\_device\_get\_bool\_property() : **Device.h**
- ob\_device\_get\_bool\_property\_range() : **Device.h**
- ob\_device\_get\_calibration\_camera\_param\_list() : **Device.h**
- ob\_device\_get\_current\_depth\_work\_mode() : **Advanced.h**
- ob\_device\_get\_current\_depth\_work\_mode\_name() : **Advanced.h**
- ob\_device\_get\_current\_preset\_name() : **Advanced.h**
- ob\_device\_get\_depth\_work\_mode\_list() : **Advanced.h**
- ob\_device\_get\_device\_info() : **Device.h**
- ob\_device\_get\_device\_state() : **Device.h**
- ob\_device\_get\_extension\_info() : **Device.h**
- ob\_device\_get\_float\_property() : **Device.h**
- ob\_device\_get\_float\_property\_range() : **Device.h**
- ob\_device\_get\_int\_property() : **Device.h**
- ob\_device\_get\_int\_property\_range() : **Device.h**
- ob\_device\_get\_multi\_device\_sync\_config() : **MultipleDevices.h**
- ob\_device\_get\_raw\_data() : **Device.h**
- ob\_device\_get\_sensor() : **Device.h**
- ob\_device\_get\_sensor\_list() : **Device.h**
- ob\_device\_get\_structured\_data() : **Device.h**
- ob\_device\_get\_supported\_multi\_device\_sync\_mode\_bitmap() : **MultipleDevices.h**
- ob\_device\_get\_supported\_property\_count() : **Device.h**
- ob\_device\_get\_supported\_property\_item() : **Device.h**
- ob\_device\_get\_timestamp\_reset\_config() : **MultipleDevices.h**
- ob\_device\_info\_get\_asicName() : **Device.h**
- ob\_device\_info\_get\_connection\_type() : **Device.h**
- ob\_device\_info\_get\_device\_type() : **Device.h**
- ob\_device\_info\_get\_firmware\_version() : **Device.h**
- ob\_device\_info\_get\_hardware\_version() : **Device.h**
- ob\_device\_info\_get\_ip\_address() : **Device.h**
- ob\_device\_info\_get\_name() : **Device.h**
- ob\_device\_info\_get\_pid() : **Device.h**
- ob\_device\_info\_get\_serial\_number() : **Device.h**
- ob\_device\_info\_get\_supported\_min\_sdk\_version() : **Device.h**
- ob\_device\_info\_get\_uid() : **Device.h**
- ob\_device\_info\_get\_vid() : **Device.h**
- ob\_device\_is\_extension\_info\_exist() : **Device.h**
- ob\_device\_is\_frame\_interleave\_supported() : **Advanced.h**
- ob\_device\_is\_global\_timestamp\_supported() : **Device.h**
- ob\_device\_is\_property\_supported() : **Device.h**
- ob\_device\_list\_get\_count() : **Device.h**
- ob\_device\_list\_get\_device() : **Device.h**
- ob\_device\_list\_get\_device\_by\_serial\_number() : **Device.h**
- ob\_device\_list\_get\_device\_by\_uid() : **Device.h**
- ob\_device\_list\_get\_device\_connection\_type() : **Device.h**

- ob\_device\_list\_get\_device\_ip\_address() : **Device.h**
- ob\_device\_list\_get\_device\_local\_mac() : **Device.h**
- ob\_device\_list\_get\_device\_name() : **Device.h**
- ob\_device\_list\_get\_device\_pid() : **Device.h**
- ob\_device\_list\_get\_device\_serial\_number() : **Device.h**
- ob\_device\_list\_get\_device\_uid() : **Device.h**
- ob\_device\_list\_get\_device\_vid() : **Device.h**
- ob\_device\_load\_frame\_interleave() : **Advanced.h**
- ob\_device\_load\_preset() : **Advanced.h**
- ob\_device\_load\_preset\_from\_json\_data() : **Advanced.h**
- ob\_device\_load\_preset\_from\_json\_file() : **Advanced.h**
- ob\_device\_preset\_list\_get\_count() : **Advanced.h**
- ob\_device\_preset\_list\_get\_name() : **Advanced.h**
- ob\_device\_preset\_list\_has\_preset() : **Advanced.h**
- ob\_device\_preset\_resolution\_config\_get\_count() : **Advanced.h**
- ob\_device\_preset\_resolution\_config\_list\_get\_item() : **Advanced.h**
- ob\_device\_read\_customer\_data() : **Device.h**
- ob\_device\_reboot() : **Device.h**
- ob\_device\_send\_and\_receive\_data() : **Device.h**
- ob\_device\_set\_bool\_property() : **Device.h**
- ob\_device\_set\_float\_property() : **Device.h**
- ob\_device\_set\_int\_property() : **Device.h**
- ob\_device\_set\_multi\_device\_sync\_config() : **MultipleDevices.h**
- ob\_device\_set\_state\_changed\_callback() : **Device.h**
- ob\_device\_set\_structured\_data() : **Device.h**
- ob\_device\_set\_timestamp\_reset\_config() : **MultipleDevices.h**
- ob\_device\_switch\_depth\_work\_mode() : **Advanced.h**
- ob\_device\_switch\_depth\_work\_mode\_by\_name() : **Advanced.h**
- ob\_device\_timer\_sync\_with\_host() : **MultipleDevices.h**
- ob\_device\_timestamp\_reset() : **MultipleDevices.h**
- ob\_device\_trigger\_capture() : **MultipleDevices.h**
- ob\_device\_update\_firmware() : **Device.h**
- ob\_device\_update\_firmware\_from\_data() : **Device.h**
- ob\_device\_update\_optional\_depth\_presets() : **Device.h**
- ob\_device\_write\_customer\_data() : **Device.h**
- ob\_disparity\_based\_stream\_profile\_get\_disparity\_param() : **StreamProfile.h**
- ob\_disparity\_based\_stream\_profile\_set\_disparity\_param() : **StreamProfile.h**
- ob\_enable\_device\_clock\_sync() : **Context.h**
- ob\_enable\_net\_device\_enumeration() : **Context.h**
- ob\_error\_get\_args() : **Error.h**
- ob\_error\_get\_exception\_type() : **Error.h**
- ob\_error\_get\_function() : **Error.h**
- ob\_error\_get\_message() : **Error.h**
- ob\_error\_get\_status() : **Error.h**

- ob\_filter\_config\_schema\_list\_get\_count() : **Filter.h**
- ob\_filter\_config\_schema\_list\_get\_item() : **Filter.h**
- ob\_filter\_enable() : **Filter.h**
- ob\_filter\_get\_config\_schema() : **Filter.h**
- ob\_filter\_get\_config\_schema\_list() : **Filter.h**
- ob\_filter\_get\_config\_value() : **Filter.h**
- ob\_filter\_get\_name() : **Filter.h**
- ob\_filter\_get\_vendor\_specific\_code() : **Filter.h**
- ob\_filter\_is\_enabled() : **Filter.h**
- ob\_filter\_list\_get\_count() : **Filter.h**
- ob\_filter\_list\_get\_filter() : **Filter.h**
- ob\_filter\_process() : **Filter.h**
- ob\_filter\_push\_frame() : **Filter.h**
- ob\_filter\_reset() : **Filter.h**
- ob\_filter\_set\_callback() : **Filter.h**
- ob\_filter\_set\_config\_value() : **Filter.h**
- ob\_filter\_update\_config() : **Filter.h**
- ob\_format\_to\_string() : **TypeHelper.h**
- ob\_format\_type\_to\_string() : **TypeHelper.h**
- ob\_frame\_add\_ref() : **Frame.h**
- ob\_frame\_copy\_info() : **Frame.h**
- ob\_frame\_get\_data() : **Frame.h**
- ob\_frame\_get\_data\_size() : **Frame.h**
- ob\_frame\_get\_device() : **Frame.h**
- ob\_frame\_get\_format() : **Frame.h**
- ob\_frame\_get\_global\_timestamp\_us() : **Frame.h**
- ob\_frame\_get\_index() : **Frame.h**
- ob\_frame\_get\_metadata() : **Frame.h**
- ob\_frame\_get\_metadata\_size() : **Frame.h**
- ob\_frame\_get\_metadata\_value() : **Frame.h**
- ob\_frame\_get\_sensor() : **Frame.h**
- ob\_frame\_get\_stream\_profile() : **Frame.h**
- ob\_frame\_get\_system\_timestamp\_us() : **Frame.h**
- ob\_frame\_get\_timestamp\_us() : **Frame.h**
- ob\_frame\_get\_type() : **Frame.h**
- ob\_frame\_has\_metadata() : **Frame.h**
- ob\_frame\_set\_stream\_profile() : **Frame.h**
- ob\_frame\_set\_system\_timestamp\_us() : **Frame.h**
- ob\_frame\_set\_timestamp\_us() : **Frame.h**
- ob\_frame\_type\_to\_string() : **TypeHelper.h**
- ob\_frame\_update\_data() : **Frame.h**
- ob\_frame\_update\_metadata() : **Frame.h**
- ob\_frameset\_get\_color\_frame() : **Frame.h**
- ob\_frameset\_get\_count() : **Frame.h**

- ob\_frameset\_get\_depth\_frame() : **Frame.h**
- ob\_frameset\_get\_frame() : **Frame.h**
- ob\_frameset\_get\_frame\_by\_index() : **Frame.h**
- ob\_frameset\_get\_ir\_frame() : **Frame.h**
- ob\_frameset\_get\_points\_frame() : **Frame.h**
- ob\_frameset\_push\_frame() : **Frame.h**
- ob\_free\_idle\_memory() : **Context.h**
- ob\_get\_d2c\_depth\_profile\_list() : **Pipeline.h**
- ob\_get\_major\_version() : **Version.h**
- ob\_get\_minor\_version() : **Version.h**
- ob\_get\_patch\_version() : **Version.h**
- ob\_get\_stage\_version() : **Version.h**
- ob\_get\_version() : **Version.h**
- ob\_gyro\_frame\_get\_temperature() : **Frame.h**
- ob\_gyro\_frame\_get\_value() : **Frame.h**
- ob\_gyro\_range\_type\_to\_string() : **TypeHelper.h**
- ob\_gyro\_stream\_get\_intrinsic() : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_get\_full\_scale\_range() : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_get\_sample\_rate() : **StreamProfile.h**
- ob\_gyro\_stream\_set\_intrinsic() : **StreamProfile.h**
- ob\_imu\_rate\_type\_to\_string() : **TypeHelper.h**
- ob\_ir\_frame\_get\_source\_sensor\_type() : **Frame.h**
- ob\_meta\_data\_type\_to\_string() : **TypeHelper.h**
- ob\_pipeline\_disable\_frame\_sync() : **Pipeline.h**
- ob\_pipeline\_enable\_frame\_sync() : **Pipeline.h**
- ob\_pipeline\_get\_calibration\_param() : **Pipeline.h**
- ob\_pipeline\_get\_camera\_param() : **Pipeline.h**
- ob\_pipeline\_get\_camera\_param\_with\_profile() : **Pipeline.h**
- ob\_pipeline\_get\_config() : **Pipeline.h**
- ob\_pipeline\_get\_device() : **Pipeline.h**
- ob\_pipeline\_get\_stream\_profile\_list() : **Pipeline.h**
- ob\_pipeline\_start() : **Pipeline.h**
- ob\_pipeline\_start\_with\_callback() : **Pipeline.h**
- ob\_pipeline\_start\_with\_config() : **Pipeline.h**
- ob\_pipeline\_stop() : **Pipeline.h**
- ob\_pipeline\_switch\_config() : **Pipeline.h**
- ob\_pipeline\_wait\_for\_frameset() : **Pipeline.h**
- ob\_playback\_device\_get\_current\_playback\_status() : **RecordPlayback.h**
- ob\_playback\_device\_get\_duration() : **RecordPlayback.h**
- ob\_playback\_device\_get\_position() : **RecordPlayback.h**
- ob\_playback\_device\_pause() : **RecordPlayback.h**
- ob\_playback\_device\_resume() : **RecordPlayback.h**
- ob\_playback\_device\_seek() : **RecordPlayback.h**
- ob\_playback\_device\_set\_playback\_rate() : **RecordPlayback.h**

- ob\_playback\_device\_set\_playback\_status\_changed\_callback() : [RecordPlayback.h](#)
- ob\_point\_cloud\_frame\_get\_height() : [Frame.h](#)
- ob\_point\_cloud\_frame\_get\_width() : [Frame.h](#)
- ob\_points\_frame\_get\_coordinate\_value\_scale() : [Frame.h](#)
- ob\_query\_device\_list() : [Context.h](#)
- ob\_record\_device\_pause() : [RecordPlayback.h](#)
- ob\_record\_device\_resume() : [RecordPlayback.h](#)
- ob\_save\_pointcloud\_to\_ply() : [Utils.h](#)
- ob\_sensor\_create\_recommended\_filter\_list() : [Sensor.h](#)
- ob\_sensor\_get\_stream\_profile\_list() : [Sensor.h](#)
- ob\_sensor\_get\_type() : [Sensor.h](#)
- ob\_sensor\_list\_get\_count() : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor() : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor\_by\_type() : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor\_type() : [Sensor.h](#)
- ob\_sensor\_start() : [Sensor.h](#)
- ob\_sensor\_stop() : [Sensor.h](#)
- ob\_sensor\_switch\_profile() : [Sensor.h](#)
- ob\_sensor\_type\_to\_stream\_type() : [TypeHelper.h](#)
- ob\_sensor\_type\_to\_string() : [TypeHelper.h](#)
- ob\_set\_device\_changed\_callback() : [Context.h](#)
- ob\_set\_extensions\_directory() : [Context.h](#)
- ob\_set\_logger\_severity() : [Context.h](#)
- ob\_set\_logger\_to\_callback() : [Context.h](#)
- ob\_set\_logger\_to\_console() : [Context.h](#)
- ob\_set\_logger\_to\_file() : [Context.h](#)
- ob\_set\_uvc\_backend\_type() : [Context.h](#)
- ob\_stream\_profile\_get\_extrinsic\_to() : [StreamProfile.h](#)
- ob\_stream\_profile\_get\_format() : [StreamProfile.h](#)
- ob\_stream\_profile\_get\_type() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_accel\_stream\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_count() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_gyro\_stream\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_video\_stream\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_extrinsic\_to() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_extrinsic\_to\_type() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_format() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_type() : [StreamProfile.h](#)
- ob\_stream\_type\_to\_string() : [TypeHelper.h](#)
- ob\_transformation\_2d\_to\_2d() : [Utils.h](#)
- ob\_transformation\_2d\_to\_3d() : [Utils.h](#)
- ob\_transformation\_3d\_to\_2d() : [Utils.h](#)
- ob\_transformation\_3d\_to\_3d() : [Utils.h](#)

- ob\_video\_frame\_get\_height() : **Frame.h**
- ob\_video\_frame\_get\_pixel\_available\_bit\_size() : **Frame.h**
- ob\_video\_frame\_get\_pixel\_type() : **Frame.h**
- ob\_video\_frame\_get\_width() : **Frame.h**
- ob\_video\_frame\_set\_pixel\_available\_bit\_size() : **Frame.h**
- ob\_video\_frame\_set\_pixel\_type() : **Frame.h**
- ob\_video\_stream\_profile\_get\_distortion() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_fps() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_height() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_intrinsic() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_width() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_distortion() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_height() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_intrinsic() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_width() : **StreamProfile.h**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all functions with links to the files they belong to:

- t -

- transformation\_depth\_frame\_to\_color\_camera() : [Utils.h](#)
- transformation\_depth\_to\_pointcloud() : [Utils.h](#)
- transformation\_depth\_to\_rgbd\_pointcloud() : [Utils.h](#)
- transformation\_init\_xy\_tables() : [Utils.h](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- g -

- getPositionValueScale : [Frame.hpp](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- h -

- HIGH\_PERFORMANCE\_MODE : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- i -

- IS\_FIXED\_SIZE\_FORMAT : [ObTypes.h](#)
- is\_ir\_frame : [ObTypes.h](#)
- is\_ir\_sensor : [ObTypes.h](#)
- is\_ir\_stream : [ObTypes.h](#)
- IS\_PACKED\_FORMAT : [ObTypes.h](#)
- isIRFrame : [ObTypes.h](#)
- isIRSensor : [ObTypes.h](#)
- isIRStream : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- 0 -

- ob\_accel\_frame\_get\_temperature() : [Frame.h](#)
- ob\_accel\_frame\_get\_value() : [Frame.h](#)
- ob\_accel\_frame\_temperature : [Frame.h](#)
- ob\_accel\_frame\_value : [Frame.h](#)
- OB\_ACCEL\_FS\_12g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_16g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_24g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_2g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_3g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_4g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_6g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_8g : [ObTypes.h](#)
- OB\_ACCEL\_FS\_UNKNOWN : [ObTypes.h](#)
- OB\_ACCEL\_FULL\_SCALE\_RANGE : [ObTypes.h](#)
- ob\_accel\_full\_scale\_range : [ObTypes.h](#)
- OB\_ACCEL\_FULL\_SCALE\_RANGE\_ANY : [ObTypes.h](#)
- ob\_accel\_intrinsic : [ObTypes.h](#)
- ob\_accel\_range\_type\_to\_string() : [TypeHelper.h](#)
- ob\_accel\_sample\_rate : [ObTypes.h](#)
- OB\_ACCEL\_SAMPLE\_RATE\_ANY : [ObTypes.h](#)
- ob\_accel\_stream\_profile\_full\_scale\_range : [StreamProfile.h](#)
- ob\_accel\_stream\_profile\_get\_full\_scale\_range() : [StreamProfile.h](#)
- ob\_accel\_stream\_profile\_get\_intrinsic() : [StreamProfile.h](#)
- ob\_accel\_stream\_profile\_get\_sample\_rate() : [StreamProfile.h](#)
- ob\_accel\_stream\_profile\_sample\_rate : [StreamProfile.h](#)
- ob\_accel\_stream\_profile\_set\_intrinsic() : [StreamProfile.h](#)
- ob\_accel\_value : [ObTypes.h](#)
- ob\_align\_filter\_set\_align\_to\_stream\_profile() : [Filter.h](#)
- ob\_align\_mode : [ObTypes.h](#)
- ob\_baseline\_calibration\_param : [ObTypes.h](#)
- OB\_BOOL\_PROPERTY : [Property.h](#)
- ob\_bool\_property\_range : [ObTypes.h](#)
- ob\_calibration\_2d\_to\_2d() : [Utils.h](#)
- ob\_calibration\_2d\_to\_3d() : [Utils.h](#)
- ob\_calibration\_3d\_to\_2d() : [Utils.h](#)
- ob\_calibration\_3d\_to\_3d() : [Utils.h](#)
- ob\_calibration\_param : [ObTypes.h](#)
- ob\_camera\_distortion : [ObTypes.h](#)
- ob\_camera\_distortion\_model : [ObTypes.h](#)
- ob\_camera\_intrinsic : [ObTypes.h](#)
- ob\_camera\_param : [ObTypes.h](#)

- ob\_camera\_param\_list : **ObTypes.h**
- ob\_camera\_param\_list\_count : **Device.h**
- ob\_camera\_param\_list\_get\_count() : **Device.h**
- ob\_camera\_param\_list\_get\_param() : **Device.h**
- ob\_camera\_performance\_mode : **ObTypes.h**
- OB\_CMD\_VERSION : **ObTypes.h**
- ob\_cmd\_version : **ObTypes.h**
- OB\_CMD\_VERSION\_INVALID : **ObTypes.h**
- OB\_CMD\_VERSION\_NOVERSION : **ObTypes.h**
- OB\_CMD\_VERSION\_V0 : **ObTypes.h**
- OB\_CMD\_VERSION\_V1 : **ObTypes.h**
- OB\_CMD\_VERSION\_V2 : **ObTypes.h**
- OB\_CMD\_VERSION\_V3 : **ObTypes.h**
- ob\_color\_point : **ObTypes.h**
- OB\_COMM\_NET : **ObTypes.h**
- OB\_COMM\_USB : **ObTypes.h**
- OB\_COMMUNICATION\_TYPE : **ObTypes.h**
- ob\_communication\_type : **ObTypes.h**
- OB\_COMPRESSION\_LOSSLESS : **ObTypes.h**
- OB\_COMPRESSION\_LOSSY : **ObTypes.h**
- OB\_COMPRESSION\_MODE : **ObTypes.h**
- ob\_compression\_mode : **ObTypes.h**
- OB\_COMPRESSION\_PARAMS : **ObTypes.h**
- ob\_compression\_params : **ObTypes.h**
- ob\_config : **ObTypes.h**
- ob\_config\_disable\_all\_stream() : **Pipeline.h**
- ob\_config\_disable\_stream() : **Pipeline.h**
- ob\_config\_enable\_accel\_stream() : **Pipeline.h**
- ob\_config\_enable\_all\_stream() : **Pipeline.h**
- ob\_config\_enable\_gyro\_stream() : **Pipeline.h**
- ob\_config\_enable\_stream() : **Pipeline.h**
- ob\_config\_enable\_stream\_with\_stream\_profile() : **Pipeline.h**
- ob\_config\_enable\_video\_stream() : **Pipeline.h**
- ob\_config\_get\_enabled\_stream\_profile\_list() : **Pipeline.h**
- ob\_config\_set\_align\_mode() : **Pipeline.h**
- ob\_config\_set\_depth\_scale\_after\_align\_require() : **Pipeline.h**
- ob\_config\_set\_depth\_scale\_require : **Pipeline.h**
- ob\_config\_set\_frame\_aggregate\_output\_mode() : **Pipeline.h**
- ob\_context : **ObTypes.h**
- ob\_convert\_format : **ObTypes.h**
- OB\_COORDINATE\_SYSTEM\_TYPE : **ObTypes.h**
- ob\_coordinate\_system\_type : **ObTypes.h**
- ob\_create\_accel\_stream\_profile() : **StreamProfile.h**
- ob\_create\_config() : **Pipeline.h**

- ob\_create\_context() : **Context.h**
- ob\_create\_context\_with\_config() : **Context.h**
- ob\_create\_error() : **Error.h**
- ob\_create\_filter() : **Filter.h**
- ob\_create\_frame() : **Frame.h**
- ob\_create\_frame\_from\_buffer() : **Frame.h**
- ob\_create\_frame\_from\_other\_frame() : **Frame.h**
- ob\_create\_frame\_from\_stream\_profile() : **Frame.h**
- ob\_create\_frameset() : **Frame.h**
- ob\_create\_gyro\_stream\_profile() : **StreamProfile.h**
- ob\_create\_net\_device() : **Context.h**
- ob\_create\_pipeline() : **Pipeline.h**
- ob\_create\_pipeline\_with\_device() : **Pipeline.h**
- ob\_create\_playback\_device() : **RecordPlayback.h**
- ob\_create\_private\_filter() : **Filter.h**
- ob\_create\_record\_device() : **RecordPlayback.h**
- ob\_create\_stream\_profile() : **StreamProfile.h**
- ob\_create\_stream\_profile\_from\_other\_stream\_profile() : **StreamProfile.h**
- ob\_create\_stream\_profile\_with\_new\_format() : **StreamProfile.h**
- ob\_create\_video\_frame() : **Frame.h**
- ob\_create\_video\_frame\_from\_buffer() : **Frame.h**
- ob\_create\_video\_stream\_profile() : **StreamProfile.h**
- OB\_CUSTOM\_DEPTH\_WORK\_MODE : **ObTypes.h**
- ob\_d2c\_transform : **ObTypes.h**
- ob\_data\_chunk : **ObTypes.h**
- ob\_data\_tran\_state : **ObTypes.h**
- OB\_DC\_POWER\_NO\_PLUGIN : **ObTypes.h**
- OB\_DC\_POWER\_PLUGIN : **ObTypes.h**
- ob\_dc\_power\_state : **ObTypes.h**
- OB\_DDO\_NOISE\_REMOVAL\_TYPE : **ObTypes.h**
- ob\_ddo\_noise\_removal\_type : **ObTypes.h**
- OB\_DEFAULT\_DECRYPT\_KEY : **ObTypes.h**
- OB\_DEFAULT\_STRIDE\_BYTES : **ObTypes.h**
- ob\_delete\_camera\_param\_list() : **Device.h**
- ob\_delete\_config() : **Pipeline.h**
- ob\_delete\_context() : **Context.h**
- ob\_delete\_depth\_work\_mode\_list() : **Advanced.h**
- ob\_delete\_device() : **Device.h**
- ob\_delete\_device\_info() : **Device.h**
- ob\_delete\_device\_list() : **Device.h**
- ob\_delete\_error() : **Error.h**
- ob\_delete\_filter() : **Filter.h**
- ob\_delete\_filter\_config\_schema\_list() : **Filter.h**
- ob\_delete\_filter\_list() : **Filter.h**

- ob\_delete\_frame() : [Frame.h](#)
- ob\_delete\_frame\_interleave\_list() : [Advanced.h](#)
- ob\_delete\_pipeline() : [Pipeline.h](#)
- ob\_delete\_preset\_list() : [Advanced.h](#)
- ob\_delete\_preset\_resolution\_config\_list() : [Advanced.h](#)
- ob\_delete\_record\_device() : [RecordPlayback.h](#)
- ob\_delete\_sensor() : [Sensor.h](#)
- ob\_delete\_sensor\_list() : [Sensor.h](#)
- ob\_delete\_stream\_profile() : [StreamProfile.h](#)
- ob\_delete\_stream\_profile\_list() : [StreamProfile.h](#)
- OB\_DEPRECATED : [Export.h](#)
- OB\_DEPRECATED\_EXPORT : [Export.h](#)
- OB\_DEPRECATED\_NO\_EXPORT : [Export.h](#)
- OB\_DEPTH\_CROPPING\_MODE : [ObTypes.h](#)
- ob\_depth\_cropping\_mode : [ObTypes.h](#)
- ob\_depth\_frame\_get\_value\_scale() : [Frame.h](#)
- ob\_depth\_frame\_set\_value\_scale() : [Frame.h](#)
- OB\_DEPTH\_PRECISION\_LEVEL : [ObTypes.h](#)
- ob\_depth\_precision\_level : [ObTypes.h](#)
- ob\_depth\_unit : [ObTypes.h](#)
- ob\_depth\_work\_mode : [ObTypes.h](#)
- ob\_depth\_work\_mode\_list : [ObTypes.h](#)
- ob\_depth\_work\_mode\_list\_count : [Advanced.h](#)
- ob\_depth\_work\_mode\_list\_get\_count() : [Advanced.h](#)
- ob\_depth\_work\_mode\_list\_get\_item() : [Advanced.h](#)
- ob\_depth\_work\_mode\_tag : [ObTypes.h](#)
- OB\_DEVELOPER\_MODE : [ObTypes.h](#)
- ob\_device : [ObTypes.h](#)
- OB\_DEVICE\_AUTO\_CAPTURE\_ENABLE\_BOOL : [Property.h](#)
- OB\_DEVICE\_AUTO\_CAPTURE\_INTERVAL\_TIME\_INT : [Property.h](#)
- ob\_device\_changed\_callback : [ObTypes.h](#)
- OB\_DEVICE\_DEPTH\_WORK\_MODE : [ObTypes.h](#)
- OB\_DEVICE DEVELOPMENT MODE : [ObTypes.h](#)
- ob\_device\_development\_mode : [ObTypes.h](#)
- ob\_device\_enable\_global\_timestamp() : [Device.h](#)
- ob\_device\_enable\_heartbeat() : [Device.h](#)
- ob\_device\_export\_current\_settings\_as\_preset\_json\_data() : [Advanced.h](#)
- ob\_device\_export\_current\_settings\_as\_preset\_json\_file() : [Advanced.h](#)
- ob\_device\_frame\_interleave\_list : [ObTypes.h](#)
- ob\_device\_frame\_interleave\_list\_get\_count() : [Advanced.h](#)
- ob\_device\_frame\_interleave\_list\_get\_name() : [Advanced.h](#)
- ob\_device\_frame\_interleave\_list\_has\_frame\_interleave() : [Advanced.h](#)
- ob\_device\_fw\_update\_callback : [ObTypes.h](#)
- ob\_device\_get\_available\_frame\_interleave\_list() : [Advanced.h](#)

- ob\_device\_get\_available\_preset\_list() : [Advanced.h](#)
- ob\_device\_get\_available\_preset\_resolution\_config\_list() : [Advanced.h](#)
- ob\_device\_get\_bool\_property() : [Device.h](#)
- ob\_device\_get\_bool\_property\_range() : [Device.h](#)
- ob\_device\_get\_calibration\_camera\_param\_list() : [Device.h](#)
- ob\_device\_get\_current\_depth\_work\_mode() : [Advanced.h](#)
- ob\_device\_get\_current\_depth\_work\_mode\_name() : [Advanced.h](#)
- ob\_device\_get\_current\_preset\_name() : [Advanced.h](#)
- ob\_device\_get\_depth\_work\_mode\_list() : [Advanced.h](#)
- ob\_device\_get\_device\_info() : [Device.h](#)
- ob\_device\_get\_device\_state() : [Device.h](#)
- ob\_device\_get\_extension\_info() : [Device.h](#)
- ob\_device\_get\_float\_property() : [Device.h](#)
- ob\_device\_get\_float\_property\_range() : [Device.h](#)
- ob\_device\_get\_int\_property() : [Device.h](#)
- ob\_device\_get\_int\_property\_range() : [Device.h](#)
- ob\_device\_get\_multi\_device\_sync\_config() : [MultipleDevices.h](#)
- ob\_device\_get\_raw\_data() : [Device.h](#)
- ob\_device\_get\_sensor() : [Device.h](#)
- ob\_device\_get\_sensor\_list() : [Device.h](#)
- ob\_device\_get\_structured\_data() : [Device.h](#)
- ob\_device\_get\_supported\_multi\_device\_sync\_mode\_bitmap() : [MultipleDevices.h](#)
- ob\_device\_get\_supported\_property : [Device.h](#)
- ob\_device\_get\_supported\_property\_count() : [Device.h](#)
- ob\_device\_get\_supported\_property\_item() : [Device.h](#)
- ob\_device\_get\_timestamp\_reset\_config() : [MultipleDevices.h](#)
- ob\_device\_info : [ObTypes.h](#)
- ob\_device\_info\_asicName : [Device.h](#)
- ob\_device\_info\_connection\_type : [Device.h](#)
- ob\_device\_info\_device\_type : [Device.h](#)
- ob\_device\_info\_firmware\_version : [Device.h](#)
- ob\_device\_info\_get\_asicName() : [Device.h](#)
- ob\_device\_info\_get\_connection\_type() : [Device.h](#)
- ob\_device\_info\_get\_device\_type() : [Device.h](#)
- ob\_device\_info\_get\_firmware\_version() : [Device.h](#)
- ob\_device\_info\_get\_hardware\_version() : [Device.h](#)
- ob\_device\_info\_get\_ip\_address() : [Device.h](#)
- ob\_device\_info\_get\_name() : [Device.h](#)
- ob\_device\_info\_get\_pid() : [Device.h](#)
- ob\_device\_info\_get\_serial\_number() : [Device.h](#)
- ob\_device\_info\_get\_supported\_min\_sdk\_version() : [Device.h](#)
- ob\_device\_info\_get\_uid() : [Device.h](#)
- ob\_device\_info\_get\_vid() : [Device.h](#)
- ob\_device\_info\_hardware\_version : [Device.h](#)

- ob\_device\_info\_ip\_address : **Device.h**
- ob\_device\_info\_name : **Device.h**
- ob\_device\_info\_pid : **Device.h**
- ob\_device\_info\_serial\_number : **Device.h**
- ob\_device\_info\_supported\_min\_sdk\_version : **Device.h**
- ob\_device\_info\_uid : **Device.h**
- ob\_device\_info\_vid : **Device.h**
- ob\_device\_ip\_addr\_config : **ObTypes.h**
- ob\_device\_is\_extension\_info\_exist() : **Device.h**
- ob\_device\_is\_frame\_interleave\_supported() : **Advanced.h**
- ob\_device\_is\_global\_timestamp\_supported() : **Device.h**
- ob\_device\_is\_property\_supported() : **Device.h**
- ob\_device\_list : **ObTypes.h**
- ob\_device\_list\_device\_count : **Device.h**
- ob\_device\_list\_get\_count() : **Device.h**
- ob\_device\_list\_get\_device() : **Device.h**
- ob\_device\_list\_get\_device\_by\_serial\_number() : **Device.h**
- ob\_device\_list\_get\_device\_by\_uid() : **Device.h**
- ob\_device\_list\_get\_device\_connection\_type() : **Device.h**
- ob\_device\_list\_get\_device\_count : **Device.h**
- ob\_device\_list\_get\_device\_ip\_address() : **Device.h**
- ob\_device\_list\_get\_device\_local\_mac() : **Device.h**
- ob\_device\_list\_get\_device\_name() : **Device.h**
- ob\_device\_list\_get\_device\_pid() : **Device.h**
- ob\_device\_list\_get\_device\_serial\_number() : **Device.h**
- ob\_device\_list\_get\_device\_uid() : **Device.h**
- ob\_device\_list\_get\_device\_vid() : **Device.h**
- ob\_device\_list\_get\_extension\_info : **Device.h**
- ob\_device\_load\_frame\_interleave() : **Advanced.h**
- ob\_device\_load\_preset() : **Advanced.h**
- ob\_device\_load\_preset\_from\_json\_data() : **Advanced.h**
- ob\_device\_load\_preset\_from\_json\_file() : **Advanced.h**
- ob\_device\_log\_severity\_level : **ObTypes.h**
- ob\_device\_preset\_list : **ObTypes.h**
- ob\_device\_preset\_list\_count : **Advanced.h**
- ob\_device\_preset\_list\_get\_count() : **Advanced.h**
- ob\_device\_preset\_list\_get\_name() : **Advanced.h**
- ob\_device\_preset\_list\_has\_preset() : **Advanced.h**
- ob\_device\_preset\_resolution\_config\_get\_count() : **Advanced.h**
- ob\_device\_preset\_resolution\_config\_list\_get\_item() : **Advanced.h**
- OB\_DEVICE\_PTP\_CLOCK\_SYNC\_ENABLE\_BOOL : **Property.h**
- ob\_device\_read\_customer\_data() : **Device.h**
- ob\_device\_reboot() : **Device.h**
- ob\_device\_send\_and\_receive\_data() : **Device.h**

- ob\_device\_serial\_number : **ObTypes.h**
- ob\_device\_set\_bool\_property() : **Device.h**
- ob\_device\_set\_float\_property() : **Device.h**
- ob\_device\_set\_int\_property() : **Device.h**
- ob\_device\_set\_multi\_device\_sync\_config() : **MultipleDevices.h**
- ob\_device\_set\_state\_changed\_callback() : **Device.h**
- ob\_device\_set\_structured\_data() : **Device.h**
- ob\_device\_set\_timestamp\_reset\_config() : **MultipleDevices.h**
- ob\_device\_state : **ObTypes.h**
- ob\_device\_state\_callback : **ObTypes.h**
- ob\_device\_state\_changed : **Device.h**
- ob\_device\_switch\_depth\_work\_mode() : **Advanced.h**
- ob\_device\_switch\_depth\_work\_mode\_by\_name() : **Advanced.h**
- OB\_DEVICE\_SYNC\_CONFIG : **ObTypes.h**
- ob\_device\_sync\_config : **ObTypes.h**
- ob\_device\_temperature : **ObTypes.h**
- ob\_device\_timer\_reset : **MultipleDevices.h**
- ob\_device\_timer\_sync\_with\_host() : **MultipleDevices.h**
- ob\_device\_timestamp\_reset() : **MultipleDevices.h**
- ob\_device\_trigger\_capture() : **MultipleDevices.h**
- OB\_DEVICE\_TYPE : **ObTypes.h**
- ob\_device\_type : **ObTypes.h**
- OB\_DEVICE\_TYPE\_UNKNOWN : **ObTypes.h**
- ob\_device\_update\_firmware() : **Device.h**
- ob\_device\_update\_firmware\_from\_data() : **Device.h**
- ob\_device\_update\_optional\_depth\_presets() : **Device.h**
- ob\_device\_upgrade : **Device.h**
- ob\_device\_upgrade\_from\_data : **Device.h**
- ob\_device\_write\_customer\_data() : **Device.h**
- ob\_disp\_offset\_config : **ObTypes.h**
- ob\_disparity\_based\_stream\_profile\_get\_disparity\_param() : **StreamProfile.h**
- ob\_disparity\_based\_stream\_profile\_set\_disparity\_param() : **StreamProfile.h**
- ob\_disparity\_param : **ObTypes.h**
- OB\_DISTORTION\_BROWN\_CONRADY : **ObTypes.h**
- OB\_DISTORTION\_BROWN\_CONRADY\_K6 : **ObTypes.h**
- OB\_DISTORTION\_INVERSE\_BROWN\_CONRADY : **ObTypes.h**
- OB\_DISTORTION\_KANNALA\_BRANDT4 : **ObTypes.h**
- OB\_DISTORTION\_MODIFIED\_BROWN\_CONRADY : **ObTypes.h**
- OB\_DISTORTION\_NONE : **ObTypes.h**
- ob\_edge\_noise\_removal\_filter\_params : **ObTypes.h**
- OB\_EDGE\_NOISE\_REMOVAL\_TYPE : **ObTypes.h**
- ob\_edge\_noise\_removal\_type : **ObTypes.h**
- ob\_enable\_device\_clock\_sync() : **Context.h**
- ob\_enable\_multi\_device\_sync : **Context.h**

- ob\_enable\_net\_device\_enumeration() : **Context.h**
- ob\_error : **ObTypes.h**
- ob\_error\_args : **Error.h**
- ob\_error\_exception\_type : **Error.h**
- ob\_error\_function : **Error.h**
- ob\_error\_get\_args() : **Error.h**
- ob\_error\_get\_exception\_type() : **Error.h**
- ob\_error\_get\_function() : **Error.h**
- ob\_error\_get\_message() : **Error.h**
- ob\_error\_get\_status() : **Error.h**
- ob\_error\_message : **Error.h**
- ob\_error\_status : **Error.h**
- OB\_EXCEPTION\_STD\_EXCEPTION : **ObTypes.h**
- ob\_exception\_type : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_CAMERA\_DISCONNECTED : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_INVALID\_VALUE : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_IO : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_MEMORY : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_NOT\_IMPLEMENTED : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_PLATFORM : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_UNKNOWN : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_UNSUPPORTED\_OPERATION : **ObTypes.h**
- OB\_EXCEPTION\_TYPE\_WRONG\_API\_CALL\_SEQUENCE : **ObTypes.h**
- OB\_EXPORT : **Export.h**
- ob\_extrinsic : **ObTypes.h**
- ob\_file\_send\_callback : **ObTypes.h**
- ob\_file\_tran\_state : **ObTypes.h**
- ob\_filter : **ObTypes.h**
- ob\_filter\_callback : **ObTypes.h**
- ob\_filter\_config\_schema\_item : **ObTypes.h**
- ob\_filter\_config\_schema\_list : **ObTypes.h**
- ob\_filter\_config\_schema\_list\_get\_count() : **Filter.h**
- ob\_filter\_config\_schema\_list\_get\_item() : **Filter.h**
- ob\_filter\_config\_value\_type : **ObTypes.h**
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_BOOLEAN : **ObTypes.h**
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_FLOAT : **ObTypes.h**
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_INT : **ObTypes.h**
- OB\_FILTER\_CONFIG\_VALUE\_TYPE\_INVALID : **ObTypes.h**
- ob\_filter\_enable() : **Filter.h**
- ob\_filter\_get\_config\_schema() : **Filter.h**
- ob\_filter\_get\_config\_schema\_list() : **Filter.h**
- ob\_filter\_get\_config\_value() : **Filter.h**
- ob\_filter\_get\_name() : **Filter.h**
- ob\_filter\_get\_vendor\_specific\_code() : **Filter.h**

- ob\_filter\_is\_enabled() : **Filter.h**
- ob\_filter\_list : **ObTypes.h**
- ob\_filter\_list\_get\_count() : **Filter.h**
- ob\_filter\_list\_get\_filter() : **Filter.h**
- ob\_filter\_process() : **Filter.h**
- ob\_filter\_push\_frame() : **Filter.h**
- ob\_filter\_reset() : **Filter.h**
- ob\_filter\_set\_callback() : **Filter.h**
- ob\_filter\_set\_config\_value() : **Filter.h**
- ob\_filter\_update\_config() : **Filter.h**
- ob\_float\_3d : **ObTypes.h**
- OB\_FLOAT\_PROPERTY : **Property.h**
- ob\_float\_property\_range : **ObTypes.h**
- ob\_format : **ObTypes.h**
- OB\_FORMAT\_ACCEL : **ObTypes.h**
- OB\_FORMAT\_ANY : **ObTypes.h**
- OB\_FORMAT\_BA81 : **ObTypes.h**
- OB\_FORMAT\_BGR : **ObTypes.h**
- OB\_FORMAT\_BGRA : **ObTypes.h**
- OB\_FORMAT\_BYR2 : **ObTypes.h**
- OB\_FORMAT\_COMPRESSED : **ObTypes.h**
- OB\_FORMAT\_GRAY : **ObTypes.h**
- OB\_FORMAT\_GYRO : **ObTypes.h**
- OB\_FORMAT\_H264 : **ObTypes.h**
- OB\_FORMAT\_H265 : **ObTypes.h**
- OB\_FORMAT\_HEVC : **ObTypes.h**
- OB\_FORMAT\_I420 : **ObTypes.h**
- OB\_FORMAT\_MJPEG : **ObTypes.h**
- OB\_FORMAT\_MJPG : **ObTypes.h**
- OB\_FORMAT\_NV12 : **ObTypes.h**
- OB\_FORMAT\_NV21 : **ObTypes.h**
- OB\_FORMAT\_POINT : **ObTypes.h**
- OB\_FORMAT\_RGB : **ObTypes.h**
- OB\_FORMAT\_RGB888 : **ObTypes.h**
- OB\_FORMAT\_RGB\_POINT : **ObTypes.h**
- OB\_FORMAT\_RGBA : **ObTypes.h**
- OB\_FORMAT\_RLE : **ObTypes.h**
- OB\_FORMAT\_RVL : **ObTypes.h**
- OB\_FORMAT\_RW16 : **ObTypes.h**
- ob\_format\_to\_string() : **TypeHelper.h**
- ob\_format\_type\_to\_string() : **TypeHelper.h**
- OB\_FORMAT\_UNKNOWN : **ObTypes.h**
- OB\_FORMAT\_UYVY : **ObTypes.h**
- OB\_FORMAT\_Y10 : **ObTypes.h**

- OB\_FORMAT\_Y11 : [ObTypes.h](#)
- OB\_FORMAT\_Y12 : [ObTypes.h](#)
- OB\_FORMAT\_Y12C4 : [ObTypes.h](#)
- OB\_FORMAT\_Y14 : [ObTypes.h](#)
- OB\_FORMAT\_Y16 : [ObTypes.h](#)
- OB\_FORMAT\_Y8 : [ObTypes.h](#)
- OB\_FORMAT\_YUY2 : [ObTypes.h](#)
- OB\_FORMAT\_YUYV : [ObTypes.h](#)
- OB\_FORMAT\_YV12 : [ObTypes.h](#)
- OB\_FORMAT\_Z16 : [ObTypes.h](#)
- OB\_FPS\_ANY : [ObTypes.h](#)
- ob\_frame : [ObTypes.h](#)
- OB\_FRAME\_ACCEL : [ObTypes.h](#)
- ob\_frame\_add\_ref() : [Frame.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_ALL\_TYPE\_FRAME\_REQUIRE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_ANY\_SITUATION : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_COLOR\_FRAME\_REQUIRE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_DISABLE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_FULL\_FRAME\_REQUIRE : [ObTypes.h](#)
- OB\_FRAME\_AGGREGATE\_OUTPUT\_MODE : [ObTypes.h](#)
- ob\_frame\_aggregate\_output\_mode : [ObTypes.h](#)
- ob\_frame\_callback : [ObTypes.h](#)
- OB\_FRAME\_COLOR : [ObTypes.h](#)
- OB\_FRAME\_CONFIDENCE : [ObTypes.h](#)
- ob\_frame\_copy\_info() : [Frame.h](#)
- ob\_frame\_data : [Frame.h](#)
- ob\_frame\_data\_size : [Frame.h](#)
- OB\_FRAME\_DEPTH : [ObTypes.h](#)
- ob\_frame\_destroy\_callback : [ObTypes.h](#)
- ob\_frame\_format : [Frame.h](#)
- ob\_frame\_get\_data() : [Frame.h](#)
- ob\_frame\_get\_data\_size() : [Frame.h](#)
- ob\_frame\_get\_device() : [Frame.h](#)
- ob\_frame\_get\_format() : [Frame.h](#)
- ob\_frame\_get\_global\_timestamp\_us() : [Frame.h](#)
- ob\_frame\_get\_index() : [Frame.h](#)
- ob\_frame\_get\_metadata() : [Frame.h](#)
- ob\_frame\_get\_metadata\_size() : [Frame.h](#)
- ob\_frame\_get\_metadata\_value() : [Frame.h](#)
- ob\_frame\_get\_sensor() : [Frame.h](#)
- ob\_frame\_get\_stream\_profile() : [Frame.h](#)
- ob\_frame\_get\_system\_timestamp\_us() : [Frame.h](#)
- ob\_frame\_get\_timestamp\_us() : [Frame.h](#)
- ob\_frame\_get\_type() : [Frame.h](#)

- ob\_frame\_global\_time\_stamp\_us : **Frame.h**
- OB\_FRAME\_GYRO : **ObTypes.h**
- ob\_frame\_has\_metadata() : **Frame.h**
- ob\_frame\_index : **Frame.h**
- OB\_FRAME\_IR : **ObTypes.h**
- OB\_FRAME\_IR\_LEFT : **ObTypes.h**
- OB\_FRAME\_IR\_RIGHT : **ObTypes.h**
- ob\_frame\_metadata : **Frame.h**
- ob\_frame\_metadata\_size : **Frame.h**
- ob\_frame\_metadata\_type : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_ACTUAL\_FRAME\_RATE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_BOTTOM : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_LEFT : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_RIGHT : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_AE\_ROI\_TOP : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_AUTO\_EXPOSURE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_AUTO\_WHITE\_BALANCE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_BACKLIGHT\_COMPENSATION : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_BRIGHTNESS : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_CONTRAST : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_COUNT : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_DISPARITY\_SEARCH\_OFFSET : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_DISPARITY\_SEARCH\_RANGE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_EMITTER\_MODE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_EXPOSURE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_EXPOSURE\_PRIORITY : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_FRAME\_NUMBER : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_FRAME\_RATE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_GAIN : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_GAMMA : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_GPIO\_INPUT\_DATA : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_HDR\_SEQUENCE\_INDEX : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_HDR\_SEQUENCE\_NAME : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_HDR\_SEQUENCE\_SIZE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_HUE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER\_LEVEL : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_LASER\_POWER\_MODE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_LASER\_STATUS : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_LOW\_LIGHT\_COMPENSATION : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_MANUAL\_WHITE\_BALANCE : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_POWER\_LINE\_FREQUENCY : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_SATURATION : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_SENSOR\_TIMESTAMP : **ObTypes.h**

- OB\_FRAME\_METADATA\_TYPE\_SHARPNESS : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_TIMESTAMP : **ObTypes.h**
- OB\_FRAME\_METADATA\_TYPE\_WHITE\_BALANCE : **ObTypes.h**
- OB\_FRAME\_POINTS : **ObTypes.h**
- OB\_FRAME\_RAW\_PHASE : **ObTypes.h**
- OB\_FRAME\_SET : **ObTypes.h**
- ob\_frame\_set\_device\_time\_stamp : **Frame.h**
- ob\_frame\_set\_device\_time\_stamp\_us : **Frame.h**
- ob\_frame\_set\_stream\_profile() : **Frame.h**
- ob\_frame\_set\_system\_time\_stamp : **Frame.h**
- ob\_frame\_set\_system\_timestamp\_us() : **Frame.h**
- ob\_frame\_set\_timestamp\_us() : **Frame.h**
- ob\_frame\_system\_time\_stamp : **Frame.h**
- ob\_frame\_system\_time\_stamp\_us : **Frame.h**
- ob\_frame\_time\_stamp : **Frame.h**
- ob\_frame\_time\_stamp\_us : **Frame.h**
- ob\_frame\_type : **ObTypes.h**
- OB\_FRAME\_TYPE\_COUNT : **ObTypes.h**
- ob\_frame\_type\_to\_string() : **TypeHelper.h**
- OB\_FRAME\_UNKNOWN : **ObTypes.h**
- ob\_frame\_update\_data() : **Frame.h**
- ob\_frame\_update\_metadata() : **Frame.h**
- OB\_FRAME\_VIDEO : **ObTypes.h**
- ob\_frameset\_callback : **ObTypes.h**
- ob\_frameset\_color\_frame : **Frame.h**
- ob\_frameset\_depth\_frame : **Frame.h**
- ob\_frameset\_frame\_count : **Frame.h**
- ob\_frameset\_get\_color\_frame() : **Frame.h**
- ob\_frameset\_get\_count() : **Frame.h**
- ob\_frameset\_get\_depth\_frame() : **Frame.h**
- ob\_frameset\_get\_frame() : **Frame.h**
- ob\_frameset\_get\_frame\_by\_index() : **Frame.h**
- ob\_frameset\_get\_frame\_count : **Frame.h**
- ob\_frameset\_get\_ir\_frame() : **Frame.h**
- ob\_frameset\_get\_points\_frame() : **Frame.h**
- ob\_frameset\_ir\_frame : **Frame.h**
- ob\_frameset\_points\_frame : **Frame.h**
- ob\_frameset\_push\_frame() : **Frame.h**
- ob\_free\_idle\_memory() : **Context.h**
- ob\_fw\_update\_state : **ObTypes.h**
- ob\_get\_d2c\_depth\_profile\_list() : **Pipeline.h**
- ob\_get\_data\_callback : **ObTypes.h**
- ob\_get\_filter : **Filter.h**
- ob\_get\_filter\_name : **Filter.h**

- ob\_get\_major\_version() : **Version.h**
- ob\_get\_minor\_version() : **Version.h**
- ob\_get\_patch\_version() : **Version.h**
- ob\_get\_stage\_version() : **Version.h**
- ob\_get\_version() : **Version.h**
- ob\_gyro\_frame\_get\_temperature() : **Frame.h**
- ob\_gyro\_frame\_get\_value() : **Frame.h**
- ob\_gyro\_frame\_temperature : **Frame.h**
- ob\_gyro\_frame\_value : **Frame.h**
- OB\_GYRO\_FS\_1000dps : **ObTypes.h**
- OB\_GYRO\_FS\_125dps : **ObTypes.h**
- OB\_GYRO\_FS\_16dps : **ObTypes.h**
- OB\_GYRO\_FS\_2000dps : **ObTypes.h**
- OB\_GYRO\_FS\_250dps : **ObTypes.h**
- OB\_GYRO\_FS\_31dps : **ObTypes.h**
- OB\_GYRO\_FS\_400dps : **ObTypes.h**
- OB\_GYRO\_FS\_500dps : **ObTypes.h**
- OB\_GYRO\_FS\_62dps : **ObTypes.h**
- OB\_GYRO\_FS\_800dps : **ObTypes.h**
- OB\_GYRO\_FS\_UNKNOWN : **ObTypes.h**
- OB\_GYRO\_FULL\_SCALE\_RANGE : **ObTypes.h**
- ob\_gyro\_full\_scale\_range : **ObTypes.h**
- OB\_GYRO\_FULL\_SCALE\_RANGE\_ANY : **ObTypes.h**
- ob\_gyro\_intrinsic : **ObTypes.h**
- ob\_gyro\_range\_type\_to\_string() : **TypeHelper.h**
- ob\_gyro\_sample\_rate : **ObTypes.h**
- OB\_GYRO\_SAMPLE\_RATE\_ANY : **ObTypes.h**
- ob\_gyro\_stream\_get\_intrinsic() : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_full\_scale\_range : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_get\_full\_scale\_range() : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_get\_sample\_rate() : **StreamProfile.h**
- ob\_gyro\_stream\_profile\_sample\_rate : **StreamProfile.h**
- ob\_gyro\_stream\_set\_intrinsic() : **StreamProfile.h**
- ob\_gyro\_value : **ObTypes.h**
- ob\_hdr\_config : **ObTypes.h**
- OB\_HEIGHT\_ANY : **ObTypes.h**
- OB\_HOLE\_FILL\_FAREST : **ObTypes.h**
- OB\_HOLE\_FILL\_NEAREST : **ObTypes.h**
- OB\_HOLE\_FILL\_TOP : **ObTypes.h**
- ob\_hole\_filling\_mode : **ObTypes.h**
- ob\_imu\_rate\_type\_to\_string() : **TypeHelper.h**
- OB\_INT\_PROPERTY : **Property.h**
- ob\_int\_property\_range : **ObTypes.h**
- ob\_ir\_frame\_get\_source\_sensor\_type() : **Frame.h**

- ob\_is\_video\_sensor\_type : **ObTypes.h**
- ob\_is\_video\_stream\_type : **ObTypes.h**
- OB\_LEFT\_HAND\_COORDINATE\_SYSTEM : **ObTypes.h**
- ob\_log\_callback : **ObTypes.h**
- ob\_log\_severity : **ObTypes.h**
- OB\_LOG\_SEVERITY\_DEBUG : **ObTypes.h**
- OB\_LOG\_SEVERITY\_ERROR : **ObTypes.h**
- OB\_LOG\_SEVERITY\_FATAL : **ObTypes.h**
- OB\_LOG\_SEVERITY\_INFO : **ObTypes.h**
- OB\_LOG\_SEVERITY\_NONE : **ObTypes.h**
- OB\_LOG\_SEVERITY\_OFF : **ObTypes.h**
- OB\_LOG\_SEVERITY\_WARN : **ObTypes.h**
- OB\_MEDIA\_ACCEL\_STREAM : **ObTypes.h**
- OB\_MEDIA\_ALL : **ObTypes.h**
- OB\_MEDIA\_BEGIN : **ObTypes.h**
- OB\_MEDIA\_CAMERA\_PARAM : **ObTypes.h**
- OB\_MEDIA\_COLOR\_STREAM : **ObTypes.h**
- OB\_MEDIA\_DEPTH\_STREAM : **ObTypes.h**
- OB\_MEDIA\_DEVICE\_INFO : **ObTypes.h**
- OB\_MEDIA\_END : **ObTypes.h**
- OB\_MEDIA\_GYRO\_STREAM : **ObTypes.h**
- OB\_MEDIA\_IR\_LEFT\_STREAM : **ObTypes.h**
- OB\_MEDIA\_IR\_RIGHT\_STREAM : **ObTypes.h**
- OB\_MEDIA\_IR\_STREAM : **ObTypes.h**
- OB\_MEDIA\_PAUSE : **ObTypes.h**
- OB\_MEDIA\_RESUME : **ObTypes.h**
- ob\_media\_state : **ObTypes.h**
- ob\_media\_state\_callback : **ObTypes.h**
- OB\_MEDIA\_STATE\_EM : **ObTypes.h**
- OB\_MEDIA\_STREAM\_INFO : **ObTypes.h**
- OB\_MEDIA\_TYPE : **ObTypes.h**
- ob\_media\_type : **ObTypes.h**
- ob\_meta\_data\_type\_to\_string() : **TypeHelper.h**
- OB\_MG\_FILTER : **ObTypes.h**
- OB\_MGA\_FILTER : **ObTypes.h**
- OB\_MGC\_FILTER : **ObTypes.h**
- ob\_mgc\_filter\_config : **ObTypes.h**
- OB\_MGH\_FILTER : **ObTypes.h**
- ob\_multi\_device\_sync\_mode : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_FREE\_RUN : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_HARDWARE\_TRIGGERING : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_IR\_IMU\_SYNC : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_PRIMARY : **ObTypes.h**
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_SECONDARY : **ObTypes.h**

- OB\_MULTI\_DEVICE\_SYNC\_MODE\_SECONDARY\_SYNCED : [ObTypes.h](#)
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_SOFTWARE\_TRIGGERING : [ObTypes.h](#)
- OB\_MULTI\_DEVICE\_SYNC\_MODE\_STANDALONE : [ObTypes.h](#)
- ob\_net\_ip\_config : [ObTypes.h](#)
- OB\_NO\_EXPORT : [Export.h](#)
- ob\_noise\_removal\_filter\_params : [ObTypes.h](#)
- OB\_NR\_LUT : [ObTypes.h](#)
- OB\_NR\_OVERALL : [ObTypes.h](#)
- OB\_PATH\_MAX : [ObTypes.h](#)
- OB\_PERMISSION\_ANY : [ObTypes.h](#)
- OB\_PERMISSION\_DENY : [ObTypes.h](#)
- OB\_PERMISSION\_READ : [ObTypes.h](#)
- OB\_PERMISSION\_READ\_WRITE : [ObTypes.h](#)
- ob\_permission\_type : [ObTypes.h](#)
- OB\_PERMISSION\_WRITE : [ObTypes.h](#)
- ob\_pipeline : [ObTypes.h](#)
- ob\_pipeline\_disable\_frame\_sync() : [Pipeline.h](#)
- ob\_pipeline\_enable\_frame\_sync() : [Pipeline.h](#)
- ob\_pipeline\_get\_calibration\_param() : [Pipeline.h](#)
- ob\_pipeline\_get\_camera\_param() : [Pipeline.h](#)
- ob\_pipeline\_get\_camera\_param\_with\_profile() : [Pipeline.h](#)
- ob\_pipeline\_get\_config() : [Pipeline.h](#)
- ob\_pipeline\_get\_device() : [Pipeline.h](#)
- ob\_pipeline\_get\_stream\_profile\_list() : [Pipeline.h](#)
- ob\_pipeline\_start() : [Pipeline.h](#)
- ob\_pipeline\_start\_with\_callback() : [Pipeline.h](#)
- ob\_pipeline\_start\_with\_config() : [Pipeline.h](#)
- ob\_pipeline\_stop() : [Pipeline.h](#)
- ob\_pipeline\_switch\_config() : [Pipeline.h](#)
- ob\_pipeline\_wait\_for\_frameset() : [Pipeline.h](#)
- OB\_PIXEL\_DEPTH : [ObTypes.h](#)
- OB\_PIXEL\_DISPARITY : [ObTypes.h](#)
- OB\_PIXEL\_RAW\_PHASE : [ObTypes.h](#)
- OB\_PIXEL\_TOF\_DEPTH : [ObTypes.h](#)
- ob\_pixel\_type : [ObTypes.h](#)
- OB\_PIXEL\_UNKNOWN : [ObTypes.h](#)
- ob\_playback\_callback : [ObTypes.h](#)
- OB\_PLAYBACK\_COUNT : [ObTypes.h](#)
- ob\_playback\_device : [ObTypes.h](#)
- ob\_playback\_device\_get\_current\_playback\_status() : [RecordPlayback.h](#)
- ob\_playback\_device\_get\_duration() : [RecordPlayback.h](#)
- ob\_playback\_device\_get\_position() : [RecordPlayback.h](#)
- ob\_playback\_device\_pause() : [RecordPlayback.h](#)
- ob\_playback\_device\_resume() : [RecordPlayback.h](#)

- ob\_playback\_device\_seek() : [RecordPlayback.h](#)
- ob\_playback\_device\_set\_playback\_rate() : [RecordPlayback.h](#)
- ob\_playback\_device\_set\_playback\_status\_changed\_callback() : [RecordPlayback.h](#)
- OB\_PLAYBACK\_PAUSED : [ObTypes.h](#)
- OB\_PLAYBACK\_PLAYING : [ObTypes.h](#)
- ob\_playback\_status : [ObTypes.h](#)
- ob\_playback\_status\_changed\_callback : [ObTypes.h](#)
- OB\_PLAYBACK\_STOPPED : [ObTypes.h](#)
- OB\_PLAYBACK\_UNKNOWN : [ObTypes.h](#)
- ob\_point : [ObTypes.h](#)
- ob\_point2f : [ObTypes.h](#)
- ob\_point3f : [ObTypes.h](#)
- ob\_point\_cloud\_frame\_get\_height() : [Frame.h](#)
- ob\_point\_cloud\_frame\_get\_width() : [Frame.h](#)
- ob\_points\_frame\_get\_coordinate\_value\_scale() : [Frame.h](#)
- ob\_points\_frame\_get\_position\_value\_scale : [Frame.h](#)
- ob\_power\_line\_freq\_mode : [ObTypes.h](#)
- OB\_POWER\_LINE\_FREQ\_MODE\_50HZ : [ObTypes.h](#)
- OB\_POWER\_LINE\_FREQ\_MODE\_60HZ : [ObTypes.h](#)
- OB\_POWER\_LINE\_FREQ\_MODE\_CLOSE : [ObTypes.h](#)
- OB\_PRECISION\_0MM05 : [ObTypes.h](#)
- OB\_PRECISION\_0MM1 : [ObTypes.h](#)
- OB\_PRECISION\_0MM2 : [ObTypes.h](#)
- OB\_PRECISION\_0MM4 : [ObTypes.h](#)
- OB\_PRECISION\_0MM5 : [ObTypes.h](#)
- OB\_PRECISION\_0MM8 : [ObTypes.h](#)
- OB\_PRECISION\_1MM : [ObTypes.h](#)
- OB\_PRECISION\_COUNT : [ObTypes.h](#)
- OB\_PRECISION\_UNKNOWN : [ObTypes.h](#)
- ob\_preset\_resolution\_config\_list : [ObTypes.h](#)
- ob\_preset\_resolution\_ratio\_config : [ObTypes.h](#)
- OB\_PROFILE\_DEFAULT : [ObTypes.h](#)
- OB\_PROP\_ANTI\_COLLUSION\_ACTIVATION\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_BOOT\_INTO\_RECOVERY\_MODE\_BOOL : [Property.h](#)
- OB\_PROP\_BRT\_BOOL : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_FRAME\_NUMBER\_INT : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_NUMBER\_INTERVAL\_INT : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_SIGNAL\_BOOL : [Property.h](#)
- OB\_PROP\_CAPTURE\_IMAGE\_TIME\_INTERVAL\_INT : [Property.h](#)
- OB\_PROP\_CAPTURE\_INTERVAL\_MODE\_INT : [Property.h](#)
- OB\_PROP\_CHECK\_PPS\_SYNC\_IN\_SIGNAL\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_AE\_MAX\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_AUTO\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_AUTO\_EXPOSURE\_PRIORITY\_INT : [Property.h](#)

- OB\_PROP\_COLOR\_AUTO\_WHITE\_BALANCE\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_BACKLIGHT\_COMPENSATION\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_BRIGHTNESS\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_CONTRAST\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_FOCUS\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_GAMMA\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_HDR\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_HUE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_MAXIMAL\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_MAXIMAL\_SHUTTER\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_COLOR\_POWER\_LINE\_FREQUENCY\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_ROLL\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_SATURATION\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_SHARPNESS\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_SHUTTER\_INT : [Property.h](#)
- OB\_PROP\_COLOR\_WHITE\_BALANCE\_INT : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_CONFIDENCE\_STREAM\_FILTER\_THRESHOLD\_INT : [Property.h](#)
- OB\_PROP\_D2C\_PREPROCESS\_BOOL : [Property.h](#)
- OB\_PROP\_DC\_POWER\_STATE\_INT : [Property.h](#)
- OB\_PROP\_DEBUG\_ESGM\_CONFIDENCE\_FLOAT : [Property.h](#)
- OB\_PROP\_DEPTH\_ALIGN\_HARDWARE\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_ALIGN\_HARDWARE\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_AUTO\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_AUTO\_EXPOSURE\_PRIORITY\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_CROPPING\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_HOLEFILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_MAX\_DIFF\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_MAX\_SPECKLE\_SIZE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_MAX\_DIFF\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_NOISE\_REMOVAL\_FILTER\_MAX\_SPECKLE\_SIZE\_INT : [Property.h](#)

- OB\_PROP\_DEPTH\_POSTFILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_PRECISION\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_RM\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_DEPTH\_SOFT\_FILTER\_BOOL : [Property.h](#)
- OB\_PROP\_DEPTH\_UNIT\_FLEXIBLE\_ADJUSTMENT\_FLOAT : [Property.h](#)
- OB\_PROP\_DEPTH\_WITH\_CONFIDENCE\_STREAM\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_COMMUNICATION\_TYPE\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_DEVELOPMENT\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_IN\_RECOVERY\_MODE\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_PERFORMANCE\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_REBOOT\_DELAY\_INT : [Property.h](#)
- OB\_PROP\_DEVICE\_REPOWER\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_USB2\_REPEAT\_IDENTIFY\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_USB3\_REPEAT\_IDENTIFY\_BOOL : [Property.h](#)
- OB\_PROP\_DEVICE\_WORK\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DISP\_SEARCH\_OFFSET\_INT : [Property.h](#)
- OB\_PROP\_DISP\_SEARCH\_RANGE\_MODE\_INT : [Property.h](#)
- OB\_PROP\_DISPARITY\_TO\_DEPTH\_BOOL : [Property.h](#)
- OB\_PROP\_EXTERNAL\_SIGNAL\_RESET\_BOOL : [Property.h](#)
- OB\_PROP\_FAN\_WORK\_MODE\_INT : [Property.h](#)
- OB\_PROP\_FLOOD\_BOOL : [Property.h](#)
- OB\_PROP\_FLOOD\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_FRAME\_INTERLEAVE\_CONFIG\_INDEX\_INT : [Property.h](#)
- OB\_PROP\_FRAME\_INTERLEAVE\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_FRAME\_INTERLEAVE\_LASER\_PATTERN\_SYNC\_DELAY\_INT : [Property.h](#)
- OB\_PROP\_GPM\_BOOL : [Property.h](#)
- OB\_PROP\_HARDWARE\_DISTORTION\_SWITCH\_BOOL : [Property.h](#)
- OB\_PROP\_HDR\_MERGE\_BOOL : [Property.h](#)
- OB\_PROP\_HEARTBEAT\_BOOL : [Property.h](#)
- OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_HW\_NOISE\_REMOVE\_FILTER\_THRESHOLD\_FLOAT : [Property.h](#)
- OB\_PROP\_INDICATOR\_LIGHT\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_AE\_MAX\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_IR\_AUTO\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_BRIGHTNESS\_INT : [Property.h](#)
- OB\_PROP\_IR\_CHANNEL\_DATA\_SOURCE\_INT : [Property.h](#)
- OB\_PROP\_IR\_EXPOSURE\_INT : [Property.h](#)
- OB\_PROP\_IR\_FLIP\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_GAIN\_INT : [Property.h](#)
- OB\_PROP\_IR\_LONG\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RECTIFY\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RIGHT\_FLIP\_BOOL : [Property.h](#)

- OB\_PROP\_IR\_RIGHT\_MIRROR\_BOOL : [Property.h](#)
- OB\_PROP\_IR\_RIGHT\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_IR\_ROTATE\_INT : [Property.h](#)
- OB\_PROP\_IR\_SHORT\_EXPOSURE\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_ALWAYS\_ON\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_CONTROL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_CURRENT\_FLOAT : [Property.h](#)
- OB\_PROP\_LASER\_ENERGY\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_HIGH\_TEMPERATURE\_PROTECT\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_HW\_ENERGY\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_MODE\_INT : [Property.h](#)
- OB\_PROP\_LASER\_ON\_OFF\_MODE\_INT : [Property.h](#)
- OB\_PROP\_LASER\_ON\_OFF\_PATTERN\_INT : [Property.h](#)
- OB\_PROP\_LASER\_OVERCURRENT\_PROTECTION\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_LASER\_POWER\_ACTUAL\_LEVEL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_POWER\_LEVEL\_CONTROL\_INT : [Property.h](#)
- OB\_PROP\_LASER\_PULSE\_WIDTH\_INT : [Property.h](#)
- OB\_PROP\_LASER\_PULSE\_WIDTH\_PROTECTION\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_LDP\_BOOL : [Property.h](#)
- OB\_PROP\_LDP\_MEASURE\_DISTANCE\_INT : [Property.h](#)
- OB\_PROP\_LDP\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_LOW\_EXPOSURE\_LASER\_CONTROL\_BOOL : [Property.h](#)
- OB\_PROP\_MAX\_DEPTH\_INT : [Property.h](#)
- OB\_PROP\_MIN\_DEPTH\_INT : [Property.h](#)
- OB\_PROP\_NETWORK\_BANDWIDTH\_TYPE\_INT : [Property.h](#)
- OB\_PROP\_ON\_CHIP\_CALIBRATION\_ENABLE\_BOOL : [Property.h](#)
- OB\_PROP\_ON\_CHIP\_CALIBRATION\_HEALTH\_CHECK\_FLOAT : [Property.h](#)
- OB\_PROP\_RECTIFY2\_BOOL : [Property.h](#)
- OB\_PROP\_RESTORE\_FACTORY\_SETTINGS\_BOOL : [Property.h](#)
- OB\_PROP\_RGB\_CUSTOM\_CROP\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_ACCEL\_FRAME\_TRANSFORMED\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_DEPTH\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_DISPARITY\_TO\_DEPTH\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_GYRO\_FRAME\_TRANSFORMED\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_IR\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_IR\_LEFT\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SDK\_IR\_RIGHT\_FRAME\_UNPACK\_BOOL : [Property.h](#)
- OB\_PROP\_SKIP\_FRAME\_BOOL : [Property.h](#)
- OB\_PROP\_SLAVE\_DEVICE\_SYNC\_STATUS\_BOOL : [Property.h](#)
- OB\_PROP\_SWITCH\_IR\_MODE\_INT : [Property.h](#)
- OB\_PROP\_SYNC\_SIGNAL\_TRIGGER\_OUT\_BOOL : [Property.h](#)
- OB\_PROP\_TEMPERATURE\_COMPENSATION\_BOOL : [Property.h](#)
- OB\_PROP\_TIMER\_RESET\_DELAY\_US\_INT : [Property.h](#)

- OB\_PROP\_TIMER\_RESET\_ENABLE\_BOOL : **Property.h**
- OB\_PROP\_TIMER\_RESET\_SIGNAL\_BOOL : **Property.h**
- OB\_PROP\_TIMER\_RESET\_TRIGGER\_OUT\_ENABLE\_BOOL : **Property.h**
- OB\_PROP\_TIMER\_RESET\_TRIGGLE\_OUT\_ENABLE\_BOOL : **Property.h**
- OB\_PROP\_TIMESTAMP\_OFFSET\_INT : **Property.h**
- OB\_PROP\_TOF\_FILTER\_RANGE\_INT : **Property.h**
- OB\_PROP\_USB\_POWER\_STATE\_INT : **Property.h**
- OB\_PROP\_WATCHDOG\_BOOL : **Property.h**
- ob\_property\_id : **Property.h**
- ob\_property\_item : **Property.h**
- ob\_property\_type : **Property.h**
- ob\_protocol\_version : **ObTypes.h**
- ob\_query\_device\_list() : **Context.h**
- OB\_RAW\_DATA\_CAMERA\_CALIB\_JSON\_FILE : **Property.h**
- ob\_record\_device : **ObTypes.h**
- ob\_record\_device\_pause() : **RecordPlayback.h**
- ob\_record\_device\_resume() : **RecordPlayback.h**
- ob\_rect : **ObTypes.h**
- ob\_region\_of\_interest : **ObTypes.h**
- OB\_RIGHT\_HAND\_COORDINATE\_SYSTEM : **ObTypes.h**
- OB\_ROTATE\_DEGREE\_0 : **ObTypes.h**
- OB\_ROTATE\_DEGREE\_180 : **ObTypes.h**
- OB\_ROTATE\_DEGREE\_270 : **ObTypes.h**
- OB\_ROTATE\_DEGREE\_90 : **ObTypes.h**
- ob\_rotate\_degree\_type : **ObTypes.h**
- OB\_SAMPLE\_RATE : **ObTypes.h**
- OB\_SAMPLE\_RATE\_100\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_12\_5\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_16\_KHZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_1\_5625\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_1\_KHZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_200\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_25\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_2\_KHZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_32\_KHZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_3\_125\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_400\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_4\_KHZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_500\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_50\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_6\_25\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_800\_HZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_8\_KHZ : **ObTypes.h**
- OB\_SAMPLE\_RATE\_UNKNOWN : **ObTypes.h**

- ob\_save\_pointcloud\_to\_ply() : [Utils.h](#)
- ob\_sensor : [ObTypes.h](#)
- OB\_SENSOR\_ACCEL : [ObTypes.h](#)
- OB\_SENSOR\_COLOR : [ObTypes.h](#)
- OB\_SENSOR\_CONFIDENCE : [ObTypes.h](#)
- ob\_sensor\_create\_recommended\_filter\_list() : [Sensor.h](#)
- OB\_SENSOR\_DEPTH : [ObTypes.h](#)
- ob\_sensor\_get\_recommended\_filter\_list : [Sensor.h](#)
- ob\_sensor\_get\_stream\_profile\_list() : [Sensor.h](#)
- ob\_sensor\_get\_type() : [Sensor.h](#)
- OB\_SENSOR\_GYRO : [ObTypes.h](#)
- OB\_SENSOR\_IR : [ObTypes.h](#)
- OB\_SENSOR\_IR\_LEFT : [ObTypes.h](#)
- OB\_SENSOR\_IR\_RIGHT : [ObTypes.h](#)
- ob\_sensor\_list : [ObTypes.h](#)
- ob\_sensor\_list\_get\_count() : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor() : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor\_by\_type() : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor\_count : [Sensor.h](#)
- ob\_sensor\_list\_get\_sensor\_type() : [Sensor.h](#)
- OB\_SENSOR\_RAW\_PHASE : [ObTypes.h](#)
- ob\_sensor\_start() : [Sensor.h](#)
- ob\_sensor\_stop() : [Sensor.h](#)
- ob\_sensor\_switch\_profile() : [Sensor.h](#)
- ob\_sensor\_type : [ObTypes.h](#)
- OB\_SENSOR\_TYPE\_COUNT : [ObTypes.h](#)
- ob\_sensor\_type\_to\_stream\_type() : [TypeHelper.h](#)
- ob\_sensor\_type\_to\_string() : [TypeHelper.h](#)
- OB\_SENSOR\_UNKNOWN : [ObTypes.h](#)
- ob\_sequence\_id\_item : [ObTypes.h](#)
- ob\_serial\_number : [ObTypes.h](#)
- ob\_set\_data\_callback : [ObTypes.h](#)
- ob\_set\_device\_changed\_callback() : [Context.h](#)
- ob\_set\_extensions\_directory() : [Context.h](#)
- ob\_set\_logger\_callback : [Context.h](#)
- ob\_set\_logger\_severity() : [Context.h](#)
- ob\_set\_logger\_to\_callback() : [Context.h](#)
- ob\_set\_logger\_to\_console() : [Context.h](#)
- ob\_set\_logger\_to\_file() : [Context.h](#)
- ob\_set\_uvc\_backend\_type() : [Context.h](#)
- ob\_spatial\_advanced\_filter\_params : [ObTypes.h](#)
- ob\_status : [ObTypes.h](#)
- OB\_STATUS\_ERROR : [ObTypes.h](#)
- OB\_STATUS\_OK : [ObTypes.h](#)

- OB\_STREAM\_ACCEL : [ObTypes.h](#)
- OB\_STREAM\_COLOR : [ObTypes.h](#)
- OB\_STREAM\_CONFIDENCE : [ObTypes.h](#)
- OB\_STREAM\_DEPTH : [ObTypes.h](#)
- OB\_STREAM\_GYRO : [ObTypes.h](#)
- OB\_STREAM\_IR : [ObTypes.h](#)
- OB\_STREAM\_IR\_LEFT : [ObTypes.h](#)
- OB\_STREAM\_IR\_RIGHT : [ObTypes.h](#)
- ob\_stream\_profile : [ObTypes.h](#)
- ob\_stream\_profile\_format : [StreamProfile.h](#)
- ob\_stream\_profile\_get\_extrinsic\_to() : [StreamProfile.h](#)
- ob\_stream\_profile\_get\_format() : [StreamProfile.h](#)
- ob\_stream\_profile\_get\_type() : [StreamProfile.h](#)
- ob\_stream\_profile\_list : [ObTypes.h](#)
- ob\_stream\_profile\_list\_count : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_accel\_stream\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_count() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_gyro\_stream\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_list\_get\_video\_stream\_profile() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_extrinsic\_to() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_extrinsic\_to\_type() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_format() : [StreamProfile.h](#)
- ob\_stream\_profile\_set\_type() : [StreamProfile.h](#)
- ob\_stream\_profile\_type : [StreamProfile.h](#)
- OB\_STREAM\_RAW\_PHASE : [ObTypes.h](#)
- ob\_stream\_type : [ObTypes.h](#)
- OB\_STREAM\_TYPE\_COUNT : [ObTypes.h](#)
- ob\_stream\_type\_to\_string() : [TypeHelper.h](#)
- OB\_STREAM\_UNKNOWN : [ObTypes.h](#)
- OB\_STREAM\_VIDEO : [ObTypes.h](#)
- OB\_STRUCT\_ASIC\_SERIAL\_NUMBER : [Property.h](#)
- OB\_STRUCT\_BASELINE\_CALIBRATION\_PARAM : [Property.h](#)
- OB\_STRUCT\_COLOR\_AE\_ROI : [Property.h](#)
- OB\_STRUCT\_CURRENT\_DEPTH\_ALG\_MODE : [Property.h](#)
- OB\_STRUCT\_DEPTH\_AE\_ROI : [Property.h](#)
- OB\_STRUCT\_DEPTH\_HDR\_CONFIG : [Property.h](#)
- OB\_STRUCT\_DEPTH\_PRECISION\_SUPPORT\_LIST : [Property.h](#)
- OB\_STRUCT\_DEVICE\_IP\_ADDR\_CONFIG : [Property.h](#)
- OB\_STRUCT\_DEVICE\_SERIAL\_NUMBER : [Property.h](#)
- OB\_STRUCT\_DEVICE\_STATIC\_IP\_CONFIG\_RECORD : [Property.h](#)
- OB\_STRUCT\_DEVICE\_TEMPERATURE : [Property.h](#)
- OB\_STRUCT\_DEVICE\_TIME : [Property.h](#)
- OB\_STRUCT\_DISP\_OFFSET\_CONFIG : [Property.h](#)

- OB\_STRUCT\_MULTI\_DEVICE\_SYNC\_CONFIG : [Property.h](#)
- OB\_STRUCT\_PRESET\_RESOLUTION\_CONFIG : [Property.h](#)
- OB\_STRUCT\_PROPERTY : [Property.h](#)
- OB\_STRUCT\_RGB\_CROP\_ROI : [Property.h](#)
- OB\_STRUCT\_TOF\_EXPOSURE\_THRESHOLD\_CONTROL : [Property.h](#)
- OB\_STRUCTURED\_LIGHT\_BINOCULAR\_CAMERA : [ObTypes.h](#)
- OB\_STRUCTURED\_LIGHT\_MONOCULAR\_CAMERA : [ObTypes.h](#)
- OB\_SYNC\_MODE : [ObTypes.h](#)
- ob\_sync\_mode : [ObTypes.h](#)
- OB\_SYNC\_MODE\_CLOSE : [ObTypes.h](#)
- OB\_SYNC\_MODE\_IR\_IMU\_SYNC : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY\_IR\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY MCU\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_PRIMARY\_SOFT\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_SECONDARY : [ObTypes.h](#)
- OB\_SYNC\_MODE\_SECONDARY\_SOFT\_TRIGGER : [ObTypes.h](#)
- OB\_SYNC\_MODE\_STANDALONE : [ObTypes.h](#)
- OB\_SYNC\_MODE\_UNKNOWN : [ObTypes.h](#)
- OB\_TOF\_CAMERA : [ObTypes.h](#)
- ob\_tof\_exposure\_threshold\_control : [ObTypes.h](#)
- ob\_tof\_filter\_range : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE CLOSE : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_DEBUG : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_LONG : [ObTypes.h](#)
- OB\_TOF\_FILTER\_RANGE\_MIDDLE : [ObTypes.h](#)
- ob\_transform : [ObTypes.h](#)
- ob\_transformation\_2d\_to\_2d() : [Utils.h](#)
- ob\_transformation\_2d\_to\_3d() : [Utils.h](#)
- ob\_transformation\_3d\_to\_2d() : [Utils.h](#)
- ob\_transformation\_3d\_to\_3d() : [Utils.h](#)
- ob\_uint16\_property\_range : [ObTypes.h](#)
- ob\_uint8\_property\_range : [ObTypes.h](#)
- ob\_upgrade\_state : [ObTypes.h](#)
- OB\_USB\_POWER\_5V\_0A9 : [ObTypes.h](#)
- OB\_USB\_POWER\_5V\_1A5 : [ObTypes.h](#)
- OB\_USB\_POWER\_5V\_3A0 : [ObTypes.h](#)
- OB\_USB\_POWER\_NO\_PLUGIN : [ObTypes.h](#)
- ob\_usb\_power\_state : [ObTypes.h](#)
- OB\_USER\_MODE : [ObTypes.h](#)
- ob\_uvc\_backend\_type : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_AUTO : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_LIBUVC : [ObTypes.h](#)
- OB\_UVC\_BACKEND\_TYPE\_MSMF : [ObTypes.h](#)

- OB\_UVC\_BACKEND\_TYPE\_V4L2 : **ObTypes.h**
- ob\_video\_frame\_get\_height() : **Frame.h**
- ob\_video\_frame\_get\_pixel\_available\_bit\_size() : **Frame.h**
- ob\_video\_frame\_get\_pixel\_type() : **Frame.h**
- ob\_video\_frame\_get\_width() : **Frame.h**
- ob\_video\_frame\_height : **Frame.h**
- ob\_video\_frame\_metadata : **Frame.h**
- ob\_video\_frame\_metadata\_size : **Frame.h**
- ob\_video\_frame\_pixel\_available\_bit\_size : **Frame.h**
- ob\_video\_frame\_set\_pixel\_available\_bit\_size() : **Frame.h**
- ob\_video\_frame\_set\_pixel\_type() : **Frame.h**
- ob\_video\_frame\_width : **Frame.h**
- ob\_video\_stream\_profile\_fps : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_distortion() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_fps() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_height() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_intrinsic() : **StreamProfile.h**
- ob\_video\_stream\_profile\_get\_width() : **StreamProfile.h**
- ob\_video\_stream\_profile\_height : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_distortion() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_height() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_intrinsic() : **StreamProfile.h**
- ob\_video\_stream\_profile\_set\_width() : **StreamProfile.h**
- ob\_video\_stream\_width : **StreamProfile.h**
- OB\_WIDTH\_ANY : **ObTypes.h**
- ob\_xy\_tables : **ObTypes.h**
- OBAccelFullScaleRange : **ObTypes.h**
- OBAccelSampleRate : **ObTypes.h**
- OBAccelMode : **ObTypes.h**
- OBBaselineCalibrationParam : **ObTypes.h**
- OBCameraDistortionModel : **ObTypes.h**
- OBCameraPerformanceMode : **ObTypes.h**
- OBCmdVersion : **ObTypes.h**
- OBCommunicationType : **ObTypes.h**
- OBCompressionMode : **ObTypes.h**
- OBConvertFormat : **ObTypes.h**
- OBCoordinateSystemType : **ObTypes.h**
- OBDATATranState : **ObTypes.h**
- OBDCPowerState : **ObTypes.h**
- OBDDONoiseRemovalType : **ObTypes.h**
- OBDepthCroppingMode : **ObTypes.h**
- OBDepthPrecisionLevel : **ObTypes.h**
- OBDepthUnit : **ObTypes.h**
- OBDepthWorkModeTag : **ObTypes.h**

- OBDeviceDevelopmentMode : **ObTypes.h**
- OBDeviceIpAddrConfig : **ObTypes.h**
- OBDeviceLogSeverityLevel : **ObTypes.h**
- OBDeviceState : **ObTypes.h**
- OBDeviceTimestampResetConfig : **ObTypes.h**
- OBDeviceType : **ObTypes.h**
- OBEdgeNoiseRemovalType : **ObTypes.h**
- OBExceptionType : **ObTypes.h**
- OBExtrinsic : **ObTypes.h**
- OBFileTranState : **ObTypes.h**
- OBFilterConfigValueType : **ObTypes.h**
- OBFloat3D : **ObTypes.h**
- OBFormat : **ObTypes.h**
- OBFrameAggregateOutputMode : **ObTypes.h**
- OBFrameMetadataType : **ObTypes.h**
- OBFrameType : **ObTypes.h**
- OBFwUpdateState : **ObTypes.h**
- OBGyroFullScaleRange : **ObTypes.h**
- OBGyroSampleRate : **ObTypes.h**
- OBGyroValue : **ObTypes.h**
- OBHdrConfig : **ObTypes.h**
- OBHoleFillingMode : **ObTypes.h**
- OBIMUSampleRate : **ObTypes.h**
- OBLogSeverity : **ObTypes.h**
- OBMarginFilterConfig : **ObTypes.h**
- OBMediaState : **ObTypes.h**
- OBMediaType : **ObTypes.h**
- OBMultiDeviceSyncConfig : **ObTypes.h**
- OBMultiDeviceSyncMode : **ObTypes.h**
- OBPermissionType : **ObTypes.h**
- OBPixelType : **ObTypes.h**
- OBPlaybackStatus : **ObTypes.h**
- OBPoint3f : **ObTypes.h**
- OBPowerLineFreqMode : **ObTypes.h**
- OBPropertyID : **Property.h**
- OBPropertyItem : **Property.h**
- OB.PropertyType : **Property.h**
- OBRegionOfInterest : **ObTypes.h**
- OBRotateDegreeType : **ObTypes.h**
- OBSensorType : **ObTypes.h**
- OBSerialNumber : **ObTypes.h**
- OBStatus : **ObTypes.h**
- OBStreamType : **ObTypes.h**
- OBSyncMode : **ObTypes.h**

- OBTofFilterRange : [ObTypes.h](#)
- OBTransform : [ObTypes.h](#)
- OBUpgradeState : [ObTypes.h](#)
- OBUSBPowerState : [ObTypes.h](#)
- OBUvcBackendType : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- S -

- STAT\_DONE : [ObTypes.h](#)
- STAT\_DONE\_WITH\_DUPLICATES : [ObTypes.h](#)
- STAT\_FILE\_TRANSFER : [ObTypes.h](#)
- STAT\_IN\_PROGRESS : [ObTypes.h](#)
- STAT\_START : [ObTypes.h](#)
- STAT\_VERIFY\_IMAGE : [ObTypes.h](#)
- STAT\_VERIFY\_SUCCESS : [ObTypes.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all file members with links to the files they belong to:

- t -

- timerReset : [Device.hpp](#)
- TOF\_EXPOSURE\_THRESHOLD\_CONTROL : [ObTypes.h](#)
- TOF\_FILTER\_RANGE : [ObTypes.h](#)
- transformation\_depth\_frame\_to\_color\_camera() : [Utils.h](#)
- transformation\_depth\_to\_pointcloud() : [Utils.h](#)
- transformation\_depth\_to\_rgbd\_pointcloud() : [Utils.h](#)
- transformation\_init\_xy\_tables() : [Utils.h](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

Here is a list of all typedefs with links to the files they belong to:

- d -

- DEVICE\_IP\_ADDR\_CONFIG : [ObTypes.h](#)
- DEVICE\_LOG\_SEVERITY\_LEVEL : [ObTypes.h](#)
- DEVICE\_TEMPERATURE : [ObTypes.h](#)

- o -

- OB\_ACCEL\_FULL\_SCALE\_RANGE : [ObTypes.h](#)
- ob\_accel\_full\_scale\_range : [ObTypes.h](#)
- ob\_accel\_intrinsic : [ObTypes.h](#)
- ob\_accel\_sample\_rate : [ObTypes.h](#)
- ob\_accel\_value : [ObTypes.h](#)
- ob\_align\_mode : [ObTypes.h](#)
- ob\_baseline\_calibration\_param : [ObTypes.h](#)
- ob\_bool\_property\_range : [ObTypes.h](#)
- ob\_calibration\_param : [ObTypes.h](#)
- ob\_camera\_distortion : [ObTypes.h](#)
- ob\_camera\_distortion\_model : [ObTypes.h](#)
- ob\_camera\_intrinsic : [ObTypes.h](#)
- ob\_camera\_param : [ObTypes.h](#)
- ob\_camera\_param\_list : [ObTypes.h](#)
- ob\_camera\_performance\_mode : [ObTypes.h](#)
- ob\_cmd\_version : [ObTypes.h](#)
- ob\_color\_point : [ObTypes.h](#)
- OB\_COMMUNICATION\_TYPE : [ObTypes.h](#)
- ob\_communication\_type : [ObTypes.h](#)
- OB\_COMPRESSION\_MODE : [ObTypes.h](#)
- ob\_compression\_mode : [ObTypes.h](#)
- OB\_COMPRESSION\_PARAMS : [ObTypes.h](#)
- ob\_compression\_params : [ObTypes.h](#)
- ob\_config : [ObTypes.h](#)
- ob\_context : [ObTypes.h](#)
- ob\_convert\_format : [ObTypes.h](#)
- ob\_coordinate\_system\_type : [ObTypes.h](#)
- ob\_d2c\_transform : [ObTypes.h](#)
- ob\_data\_chunk : [ObTypes.h](#)
- ob\_data\_tran\_state : [ObTypes.h](#)
- ob\_dc\_power\_state : [ObTypes.h](#)
- ob\_ddo\_noise\_removal\_type : [ObTypes.h](#)
- OB\_DEPTH\_CROPPING\_MODE : [ObTypes.h](#)
- ob\_depth\_cropping\_mode : [ObTypes.h](#)
- OB\_DEPTH\_PRECISION\_LEVEL : [ObTypes.h](#)

- ob\_depth\_precision\_level : **ObTypes.h**
- ob\_depth\_unit : **ObTypes.h**
- ob\_depth\_work\_mode : **ObTypes.h**
- ob\_depth\_work\_mode\_list : **ObTypes.h**
- ob\_depth\_work\_mode\_tag : **ObTypes.h**
- ob\_device : **ObTypes.h**
- ob\_device\_changed\_callback : **ObTypes.h**
- ob\_device\_development\_mode : **ObTypes.h**
- ob\_device\_frame\_interleave\_list : **ObTypes.h**
- ob\_device\_fw\_update\_callback : **ObTypes.h**
- ob\_device\_info : **ObTypes.h**
- ob\_device\_list : **ObTypes.h**
- ob\_device\_log\_severity\_level : **ObTypes.h**
- ob\_device\_preset\_list : **ObTypes.h**
- ob\_device\_serial\_number : **ObTypes.h**
- ob\_device\_state : **ObTypes.h**
- ob\_device\_state\_callback : **ObTypes.h**
- OB\_DEVICE\_SYNC\_CONFIG : **ObTypes.h**
- ob\_device\_sync\_config : **ObTypes.h**
- ob\_device\_temperature : **ObTypes.h**
- OB\_DEVICE\_TYPE : **ObTypes.h**
- ob\_device\_type : **ObTypes.h**
- ob\_disp\_offset\_config : **ObTypes.h**
- ob\_disparity\_param : **ObTypes.h**
- ob\_edge\_noise\_removal\_filter\_params : **ObTypes.h**
- ob\_edge\_noise\_removal\_type : **ObTypes.h**
- ob\_error : **ObTypes.h**
- ob\_exception\_type : **ObTypes.h**
- ob\_extrinsic : **ObTypes.h**
- ob\_file\_send\_callback : **ObTypes.h**
- ob\_file\_tran\_state : **ObTypes.h**
- ob\_filter : **ObTypes.h**
- ob\_filter\_config\_schema\_item : **ObTypes.h**
- ob\_filter\_config\_schema\_list : **ObTypes.h**
- ob\_filter\_config\_value\_type : **ObTypes.h**
- ob\_filter\_list : **ObTypes.h**
- ob\_float\_3d : **ObTypes.h**
- ob\_float\_property\_range : **ObTypes.h**
- ob\_format : **ObTypes.h**
- ob\_frame : **ObTypes.h**
- ob\_frame\_aggregate\_output\_mode : **ObTypes.h**
- ob\_frame\_callback : **ObTypes.h**
- ob\_frame\_destroy\_callback : **ObTypes.h**
- ob\_frame\_type : **ObTypes.h**

- ob\_frameset\_callback : **ObTypes.h**
- ob\_fw\_update\_state : **ObTypes.h**
- ob\_get\_data\_callback : **ObTypes.h**
- OB\_GYRO\_FULL\_SCALE\_RANGE : **ObTypes.h**
- ob\_gyro\_full\_scale\_range : **ObTypes.h**
- ob\_gyro\_intrinsic : **ObTypes.h**
- ob\_gyro\_sample\_rate : **ObTypes.h**
- ob\_gyro\_value : **ObTypes.h**
- ob\_hdr\_config : **ObTypes.h**
- ob\_hole\_filling\_mode : **ObTypes.h**
- ob\_int\_property\_range : **ObTypes.h**
- ob\_log\_callback : **ObTypes.h**
- ob\_log\_severity : **ObTypes.h**
- ob\_media\_state : **ObTypes.h**
- ob\_media\_state\_callback : **ObTypes.h**
- OB\_MEDIA\_STATE\_EM : **ObTypes.h**
- OB\_MEDIA\_TYPE : **ObTypes.h**
- ob\_media\_type : **ObTypes.h**
- ob\_mgc\_filter\_config : **ObTypes.h**
- ob\_net\_ip\_config : **ObTypes.h**
- ob\_noise\_removal\_filter\_params : **ObTypes.h**
- ob\_permission\_type : **ObTypes.h**
- ob\_pipeline : **ObTypes.h**
- ob\_pixel\_type : **ObTypes.h**
- ob\_playback\_device : **ObTypes.h**
- ob\_playback\_status\_changed\_callback : **ObTypes.h**
- ob\_point : **ObTypes.h**
- ob\_point2f : **ObTypes.h**
- ob\_point3f : **ObTypes.h**
- ob\_preset\_resolution\_config\_list : **ObTypes.h**
- ob\_preset\_resolution\_ratio\_config : **ObTypes.h**
- ob\_property\_id : **Property.h**
- ob\_property\_item : **Property.h**
- ob\_property\_type : **Property.h**
- ob\_protocol\_version : **ObTypes.h**
- ob\_record\_device : **ObTypes.h**
- ob\_rect : **ObTypes.h**
- ob\_region\_of\_interest : **ObTypes.h**
- OB\_SAMPLE\_RATE : **ObTypes.h**
- ob\_sensor : **ObTypes.h**
- ob\_sensor\_list : **ObTypes.h**
- ob\_sensor\_type : **ObTypes.h**
- ob\_sequence\_id\_item : **ObTypes.h**
- ob\_serial\_number : **ObTypes.h**

- ob\_set\_data\_callback : **ObTypes.h**
- ob\_spatial\_advanced\_filter\_params : **ObTypes.h**
- ob\_status : **ObTypes.h**
- ob\_stream\_profile : **ObTypes.h**
- ob\_stream\_profile\_list : **ObTypes.h**
- ob\_stream\_type : **ObTypes.h**
- OB\_SYNC\_MODE : **ObTypes.h**
- ob\_sync\_mode : **ObTypes.h**
- ob\_toe\_exposure\_threshold\_control : **ObTypes.h**
- ob\_toe\_filter\_range : **ObTypes.h**
- ob\_transform : **ObTypes.h**
- ob\_uint16\_property\_range : **ObTypes.h**
- ob\_uint8\_property\_range : **ObTypes.h**
- ob\_upgrade\_state : **ObTypes.h**
- ob\_usb\_power\_state : **ObTypes.h**
- ob\_xy\_tables : **ObTypes.h**
- OBAccelSampleRate : **ObTypes.h**
- OBBaselineCalibrationParam : **ObTypes.h**
- OBCmdVersion : **ObTypes.h**
- OBCoordinateSystemType : **ObTypes.h**
- OBDDONoiseRemovalType : **ObTypes.h**
- OBDepthUnit : **ObTypes.h**
- OBDeviceDevelopmentMode : **ObTypes.h**
- OBDeviceLogSeverityLevel : **ObTypes.h**
- OBDeviceState : **ObTypes.h**
- OBDeviceTimestampResetConfig : **ObTypes.h**
- OBEedgeNoiseRemovalType : **ObTypes.h**
- OBEtrinsic : **ObTypes.h**
- OBFloat3D : **ObTypes.h**
- OBFrameAggregateOutputMode : **ObTypes.h**
- OBFrameMetadataType : **ObTypes.h**
- OBFwUpdateState : **ObTypes.h**
- OBGyroSampleRate : **ObTypes.h**
- OBGyroValue : **ObTypes.h**
- OBHdrConfig : **ObTypes.h**
- OBMarginFilterConfig : **ObTypes.h**
- OBMultiDeviceSyncConfig : **ObTypes.h**
- OBMultiDeviceSyncMode : **ObTypes.h**
- OBPlaybackStatus : **ObTypes.h**
- OBPoint3f : **ObTypes.h**
- OBPowerLineFreqMode : **ObTypes.h**
- OBPropertyItem : **Property.h**
- OB.PropertyType : **Property.h**
- OBRegionOfInterest : **ObTypes.h**

- OBRotateDegreeType : [ObTypes.h](#)
- OBSerialNumber : [ObTypes.h](#)
- OBTransform : [ObTypes.h](#)
- OBUvcBackendType : [ObTypes.h](#)

- t -

- TOF\_EXPOSURE\_THRESHOLD\_CONTROL : [ObTypes.h](#)
- TOF\_FILTER\_RANGE : [ObTypes.h](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# Class Hierarchy

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

[detail level 1 2 3 4]

C <a href="#">AE_ROI</a>	The rect of the region of interest
C <a href="#">BASELINE_CALIBRATION_PARAM</a>	Baseline calibration parameters
C <a href="#">ob::CameraParamList</a>	Class representing a list of camera parameters
C <a href="#">ob::Config</a>	<b>Config</b> class for configuring pipeline parameters
C <a href="#">ob::Context</a>	
C <a href="#">ob::CoordinateTransformHelper</a>	
C <a href="#">ob::Device</a>	
C <a href="#">ob::PlaybackDevice</a>	
C <a href="#">ob::DeviceFrameInterleaveList</a>	Class representing a list of device <b>Frame</b> Interleave
C <a href="#">ob::DeviceInfo</a>	A class describing device information, representing the name, id, serial number and other basic information of an RGBD camera
C <a href="#">ob::DeviceList</a>	Class representing a list of devices
C <a href="#">ob::DevicePresetList</a>	Class representing a list of device presets
C <a href="#">std::enable_shared_from_this</a>	
C <a href="#">ob::Filter</a>	Base class for all filters in the SDK
C <a href="#">ob::Align</a>	<b>Align</b> for depth to other or other to depth
C <a href="#">ob::DecimationFilter</a>	Decimation filter, reducing complexity by subsampling depth maps and losing depth details
C <a href="#">ob::DisparityTransform</a>	Depth to disparity or disparity to depth
C <a href="#">ob::FormatConvertFilter</a>	Subclass of <b>Filter</b> that performs format conversion
C <a href="#">ob::HdrMerge</a>	<b>HdrMerge</b> processing block, the processing merges between depth frames with different sub-preset sequence ids
C <a href="#">ob::HoleFillingFilter</a>	Hole filling filter, the processing performed depends on the selected hole filling mode
C <a href="#">ob::NoiseRemovalFilter</a>	The noise removal filter, removing scattering depth pixels
C <a href="#">ob::PointCloudFilter</a>	Subclass of <b>Filter</b> that generates point clouds
C <a href="#">ob::SequenceIdFilter</a>	Create <b>SequenceIdFilter</b> processing block

<b>C <a href="#">ob::SpatialAdvancedFilter</a></b>	Spatial advanced filte smooths the image by calculating frame with alpha and delta settings alpha defines the weight of the current pixel for smoothing, delta defines the depth gradient below which the smoothing will occur as number of depth levels
<b>C <a href="#">ob::TemporalFilter</a></b>	Temporal filter
<b>C <a href="#">ob::ThresholdFilter</a></b>	Creates depth Thresholding filter By controlling min and max options on the block
<b>C <a href="#">ob::Frame</a></b>	Define the frame class, which is the base class of all frame types
<b>C <a href="#">ob::AccelFrame</a></b>	Define the <a href="#">AccelFrame</a> class, which inherits from the <a href="#">Frame</a> class
<b>C <a href="#">ob::FrameSet</a></b>	Define the <a href="#">FrameSet</a> class, which inherits from the <a href="#">Frame</a> class
<b>C <a href="#">ob::GyroFrame</a></b>	Define the <a href="#">GyroFrame</a> class, which inherits from the <a href="#">Frame</a> class
<b>C <a href="#">ob::PointsFrame</a></b>	Define the <a href="#">PointsFrame</a> class, which inherits from the <a href="#">Frame</a> class
<b>C <a href="#">ob::VideoFrame</a></b>	Define the <a href="#">VideoFrame</a> class, which inherits from the <a href="#">Frame</a> class
<b>C <a href="#">ob::StreamProfile</a></b>	
<b>C <a href="#">ob::AccelStreamProfile</a></b>	Class representing an accelerometer stream profile
<b>C <a href="#">ob::GyroStreamProfile</a></b>	Class representing a gyroscope stream profile
<b>C <a href="#">ob::VideoStreamProfile</a></b>	Class representing a video stream profile
<b>C <a href="#">std::exception</a></b>	
<b>C <a href="#">ob::Error</a></b>	
<b>C <a href="#">ob::FilterFactory</a></b>	A factory class for creating filters
<b>C <a href="#">ob::FrameFactory</a></b>	<a href="#">FrameFactory</a> class, which provides some static functions to create frame objects
<b>C <a href="#">ob::FrameHelper</a></b>	FrameHepler class, which provides some static functions to set timestamp for frame objects
<b>C <a href="#">HDR_CONFIG</a></b>	FrameHepler inherited from the <a href="#">FrameFactory</a> and the timestamp interface implement here both for compatibility purposes
<b>C <a href="#">ob_device_timestamp_reset_config</a></b>	HDR Configuration
<b>C <a href="#">ob_error</a></b>	The timestamp reset configuration of the device
	The error class exposed by the SDK, users can get detailed error information according to the error

<b>C <a href="#">ob_margin_filter_config</a></b>	Configuration for depth margin filter
<b>C <a href="#">ob_multi_device_sync_config</a></b>	The synchronization configuration of the device
<b>C <a href="#">OBAccelIntrinsic</a></b>	Structure for accelerometer intrinsic parameters
<b>C <a href="#">OBAccelValue</a></b>	Data structures for accelerometers and gyroscopes
<b>C <a href="#">OBBoolPropertyRange</a></b>	Structure for boolean range
<b>C <a href="#">OBCalibrationParam</a></b>	Calibration parameters
<b>C <a href="#">OBCameraDistortion</a></b>	Structure for distortion parameters
<b>C <a href="#">OBCameraIntrinsic</a></b>	Structure for camera intrinsic parameters
<b>C <a href="#">OBCameraParam</a></b>	Structure for camera parameters
<b>C <a href="#">OBColorPoint</a></b>	3D point structure with color information
<b>C <a href="#">OBCompressionParams</a></b>	
<b>C <a href="#">OBD2CTransform</a></b>	Structure for rotation/transformation
<b>C <a href="#">OBDataChunk</a></b>	Structure for transmitting data blocks
<b>C <a href="#">OBDepthWorkMode</a></b>	Depth work mode
<b>C <a href="#">ob::OBDepthWorkModeList</a></b>	
<b>C <a href="#">OBDeviceSerialNumber</a></b>	Struct of serial number
<b>C <a href="#">OBDeviceSyncConfig</a></b>	Device synchronization configuration
<b>C <a href="#">OBDeviceTemperature</a></b>	Temperature parameters of the device (unit: Celsius)
<b>C <a href="#">OBDisparityParam</a></b>	Disparity parameters for disparity based camera
<b>C <a href="#">OBDispOffsetConfig</a></b>	Disparity offset interleaving configuration
<b>C <a href="#">OBEdgeNoiseRemovalFilterParams</a></b>	
<b>C <a href="#">OBFilterConfigSchemaItem</a></b>	Configuration Item for the filter
<b>C <a href="#">ob::OBFilterList</a></b>	
<b>C <a href="#">OBFloatPropertyRange</a></b>	Structure for float range
<b>C <a href="#">OBGyroIntrinsic</a></b>	Structure for gyroscope intrinsic parameters
<b>C <a href="#">OBIntPropertyRange</a></b>	Structure for integer range
<b>C <a href="#">OBMGCFilterConfig</a></b>	Configuration for mgc filter
<b>C <a href="#">OBNetIpConfig</a></b>	IP address configuration for network devices (IPv4)
<b>C <a href="#">OBNoiseRemovalFilterParams</a></b>	
<b>C <a href="#">OBPoint</a></b>	3D point structure in the SDK
<b>C <a href="#">OBPoint2f</a></b>	2D point structure in the SDK
<b>C <a href="#">OBPresetResolutionConfig</a></b>	

<b>C</b> <a href="#">OBPropertyItem</a>	Used to describe the characteristics of each property
<b>C</b> <a href="#">OBProtocolVersion</a>	Control command protocol version number
<b>C</b> <a href="#">OBRect</a>	Rectangle
<b>C</b> <a href="#">OBSequenceIdItem</a>	SequenceId fliter list item
<b>C</b> <a href="#">OBSpatialAdvancedFilterParams</a>	
<b>C</b> <a href="#">OBTofExposureThresholdControl</a>	TOF Exposure Threshold
<b>C</b> <a href="#">OBUInt16PropertyRange</a>	Structure for float range
<b>C</b> <a href="#">OBUInt8PropertyRange</a>	Structure for float range
<b>C</b> <a href="#">OBXYTables</a>	
<b>C</b> <a href="#">ob::Pipeline</a>	
<b>C</b> <a href="#">ob::PointCloudHelper</a>	
<b>C</b> <a href="#">ob::PresetResolutionConfigList</a>	Class representing a list of device <a href="#">Frame</a> Interleave
<b>C</b> <a href="#">ob::RangeTraits&lt; T &gt;</a>	Get the type of a PropertyRange member
<b>C</b> <a href="#">ob::RangeTraits&lt; OBFloatPropertyRange &gt;</a>	
<b>C</b> <a href="#">ob::RangeTraits&lt; OBIntPropertyRange &gt;</a>	
<b>C</b> <a href="#">ob::RangeTraits&lt; OBUInt16PropertyRange &gt;</a>	
<b>C</b> <a href="#">ob::RangeTraits&lt; OBUInt8PropertyRange &gt;</a>	
<b>C</b> <a href="#">ob::RecordDevice</a>	
<b>C</b> <a href="#">ob::Sensor</a>	
<b>C</b> <a href="#">ob::SensorList</a>	
<b>C</b> <a href="#">ob::StreamProfileFactory</a>	
<b>C</b> <a href="#">ob::StreamProfileList</a>	
<b>C</b> <a href="#">ob::TypeHelper</a>	
<b>C</b> <a href="#">ob::Version</a>	

# OrbbecSDK Documentation

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all namespace members with links to the namespace documentation for each member:

- FilterCallback : [ob](#)
- getPropertyRange() : [ob](#)
- PlaybackStatusChangeCallback : [ob](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

Here is a list of all namespace functions with links to the namespace documentation for each function:

- `getPropertyRange()` : **ob**

Here is a list of all namespace typedefs with links to the namespace documentation for each typedef:

- FilterCallback : [ob](#)
- PlaybackStatusChangeCallback : [ob](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# ob Namespace Reference

## Classes

class **AccelFrame**

    Define the **AccelFrame** class, which inherits from the **Frame** class. [More...](#)

class **AccelStreamProfile**

    Class representing an accelerometer stream profile. [More...](#)

class **Align**

**Align** for depth to other or other to depth. [More...](#)

class **CameraParamList**

    Class representing a list of camera parameters. [More...](#)

class **ColorFrame**

    Define the **ColorFrame** class, which inherits from the **VideoFrame** class. [More...](#)

class **ConfidenceFrame**

    Define the **ConfidenceFrame** class, which inherits from the **VideoFrame** class. [More...](#)

class **Config**

**Config** class for configuring pipeline parameters. [More...](#)

class **Context**

class **CoordinateTransformHelper**

class **DecimationFilter**

    Decimation filter, reducing complexity by subsampling depth maps and losing depth details.

[More...](#)

class **DepthFrame**

    Define the **DepthFrame** class, which inherits from the **VideoFrame** class. [More...](#)

class **Device**

class **DeviceFrameInterleaveList**

    Class representing a list of device **Frame** Interleave. [More...](#)

class **DeviceInfo**

    A class describing device information, representing the name, id, serial number and other basic information of an RGBD camera. [More...](#)

class **DeviceList**

    Class representing a list of devices. [More...](#)

class **DevicePresetList**

    Class representing a list of device presets. [More...](#)

class **DisparityTransform**

    Depth to disparity or disparity to depth. [More...](#)

- class **Error**
- class **Filter**
  - The **Filter** class is the base class for all filters in the SDK. [More...](#)
- class **FilterFactory**
  - A factory class for creating filters. [More...](#)
- class **FormatConvertFilter**
  - The **FormatConvertFilter** class is a subclass of **Filter** that performs format conversion. [More...](#)
- class **Frame**
  - Define the frame class, which is the base class of all frame types. [More...](#)
- class **FrameFactory**
  - FrameFactory** class, which provides some static functions to create frame objects. [More...](#)
- class **FrameHelper**
  - FrameHepler class, which provides some static functions to set timestamp for frame objects
  - FrameHepler inherited from the **FrameFactory** and the timestamp interface implement here both for compatibility purposes. [More...](#)
- class **FrameSet**
  - Define the **FrameSet** class, which inherits from the **Frame** class. [More...](#)
- class **GyroFrame**
  - Define the **GyroFrame** class, which inherits from the **Frame** class. [More...](#)
- class **GyroStreamProfile**
  - Class representing a gyroscope stream profile. [More...](#)
- class **HdrMerge**
  - HdrMerge** processing block, the processing merges between depth frames with different sub-preset sequence ids. [More...](#)
- class **HoleFillingFilter**
  - Hole filling filter, the processing performed depends on the selected hole filling mode. [More...](#)
- class **IRFrame**
  - Define the **IRFrame** class, which inherits from the **VideoFrame** class. [More...](#)
- class **NoiseRemovalFilter**
  - The noise removal filter, removing scattering depth pixels. [More...](#)
- class **OBDepthWorkModeList**
- class **OBFilterList**
- class **Pipeline**
- class **PlaybackDevice**
- class **PointCloudFilter**
  - The **PointCloudFilter** class is a subclass of **Filter** that generates point clouds. [More...](#)
- class **PointCloudHelper**
- class **PointsFrame**

Define the **PointsFrame** class, which inherits from the **Frame** class. [More...](#)

class **PresetResolutionConfigList**

Class representing a list of device **Frame** Interleave. [More...](#)

struct **RangeTraits**

Get the type of a PropertyRange member. [More...](#)

struct **RangeTraits< OBFloatPropertyRange >**

struct **RangeTraits< OBIntPropertyRange >**

struct **RangeTraits< OBUInt16PropertyRange >**

struct **RangeTraits< OBUInt8PropertyRange >**

class **RecordDevice**

class **Sensor**

class **SensorList**

class **SequenceIdFilter**

Create **SequenceIdFilter** processing block. [More...](#)

class **SpatialAdvancedFilter**

Spatial advanced filte smooths the image by calculating frame with alpha and delta settings alpha defines the weight of the current pixel for smoothing, delta defines the depth gradient below which the smoothing will occur as number of depth levels. [More...](#)

class **StreamProfile**

class **StreamProfileFactory**

class **StreamProfileList**

class **TemporalFilter**

Temporal filter. [More...](#)

class **ThresholdFilter**

Creates depth Thresholding filter By controlling min and max options on the block. [More...](#)

class **TypeHelper**

class **Version**

class **VideoFrame**

Define the **VideoFrame** class, which inherits from the **Frame** class. [More...](#)

class **VideoStreamProfile**

Class representing a video stream profile. [More...](#)

## Typedefs

typedef std::function< void(std::shared\_ptr< **Frame** >)> **FilterCallback**

A callback function that takes a shared pointer to a **Frame** object as its argument.

typedef std::function< void(**OBPlaybackStatus** status)> **PlaybackStatusChangeCallback**

## Functions

template<typename T>

T **getPropertyRange** (const **OBFilterConfigSchemaItem** &item, const double cur)

Get T Property Range.

## Detailed Description

**Frame** classis inheritance hierarchy: **Frame** | +-----+-----+-----+-----+ | | | | **VideoFrame**  
**PointsFrame** **AccelFrame** **GyroFrame** **FrameSet** | +---+-----+-----+-----+ | | | **ColorFrame**  
**DepthFrame** **IRFrame** | +---+---+ | | **IRLeftFrame** **IRRightFrame**

## Typedef Documentation

### ◆ FilterCallback

typedef std::function<void(std::shared\_ptr<**Frame**>)> **ob::FilterCallback**

A callback function that takes a shared pointer to a **Frame** object as its argument.

Definition at line [32](#) of file **Filter.hpp**.

### ◆ PlaybackStatusChangeCallback

typedef std::function<void(**OBPlaybackStatus** status)> **ob::PlaybackStatusChangeCallback**

Definition at line [19](#) of file **RecordPlayback.hpp**.

## Function Documentation

### ◆ getPropertyRange()

```
template<typename T>
T ob::getPropertyRange ( const OBFilterConfigSchemaItem & item,
                        const double cur )
```

Get T Property Range.

Definition at line **60** of file [Filter.hpp](#).

Referenced by [ob::SpatialAdvancedFilter::getAlphaRange\(\)](#),  
[ob::TemporalFilter::getDiffScaleRange\(\)](#), [ob::NoiseRemovalFilter::getDispDiffRange\(\)](#),  
[ob::SpatialAdvancedFilter::getDispDiffRange\(\)](#), [ob::SpatialAdvancedFilter::getMagnitudeRange\(\)](#),  
[ob::ThresholdFilter::getMaxRange\(\)](#), [ob::NoiseRemovalFilter::getMaxSizeRange\(\)](#),  
[ob::ThresholdFilter::getMinRange\(\)](#), [ob::SpatialAdvancedFilter::getRadiusRange\(\)](#),  
[ob::DecimationFilter::getScaleRange\(\)](#), and [ob::TemporalFilter::getWeightRange\(\)](#).

# Namespace List

Here is a list of all namespaces with brief descriptions:

N **ob** |

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## Related Pages

Here is a list of all related documentation pages:

[\*\*Deprecated List\*\*](#)

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## AE\_ROI Member List

This is the complete list of members for **AE\_ROI**, including all inherited members.

**x0\_left** AE\_ROI

**x1\_right** AE\_ROI

**y0\_top** AE\_ROI

**y1\_bottom** AE\_ROI

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## AE\_ROI Struct Reference

The rect of the region of interest. [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

```
int16_t x0\_left
int16_t y0\_top
int16_t x1\_right
int16_t y1\_bottom
```

### Detailed Description

The rect of the region of interest.

Definition at line [1442](#) of file [ObTypes.h](#).

### Member Data Documentation

#### ◆ [x0\\_left](#)

```
int16_t AE_ROI::x0\_left
```

Definition at line [1443](#) of file [ObTypes.h](#).

#### ◆ [y0\\_top](#)

```
int16_t AE_ROI::y0\_top
```

Definition at line [1444](#) of file [ObTypes.h](#).

#### ◆ [x1\\_right](#)

```
int16_t AE_ROI::x1_right
```

Definition at line [1445](#) of file [\*\*ObTypes.h\*\*](#).

◆ [y1\\_bottom](#)

```
int16_t AE_ROI::y1_bottom
```

Definition at line [1446](#) of file [\*\*ObTypes.h\*\*](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[\*\*ObTypes.h\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## **BASELINE\_CALIBRATION\_PARAM** Member List

This is the complete list of members for **BASELINE\_CALIBRATION\_PARAM**, including all inherited members.

**baseline** **BASELINE\_CALIBRATION\_PARAM**

**zpd** **BASELINE\_CALIBRATION\_PARAM**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# BASELINE\_CALIBRATION\_PARAM Struct Reference

Baseline calibration parameters. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

float **baseline**

Baseline length.

float **zpd**

Calibration distance.

## Detailed Description

Baseline calibration parameters.

Definition at line [1408](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **baseline**

```
float BASELINE_CALIBRATION_PARAM::baseline
```

Baseline length.

Definition at line [1412](#) of file [ObTypes.h](#).

◆ **zpd**

```
float BASELINE_CALIBRATION_PARAM::zpd
```

Calibration distance.

Definition at line [1416](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## **HDR\_CONFIG** Member List

This is the complete list of members for **HDR\_CONFIG**, including all inherited members.

<b>enable</b>	<b>HDR_CONFIG</b>
<b>exposure_1</b>	<b>HDR_CONFIG</b>
<b>exposure_2</b>	<b>HDR_CONFIG</b>
<b>gain_1</b>	<b>HDR_CONFIG</b>
<b>gain_2</b>	<b>HDR_CONFIG</b>
<b>sequence_name</b>	<b>HDR_CONFIG</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# HDR\_CONFIG Struct Reference

HDR Configuration. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint8_t enable`

Enable/disable HDR, after enabling HDR, the exposure\_1 and gain\_1 will be used as the first exposure and gain, and the exposure\_2 and gain\_2 will be used as the second exposure and gain. The output image will be alternately exposed and gain between the first and second exposure and gain.

`uint8_t sequence_name`

Sequence name.

`uint32_t exposure_1`

Exposure time 1.

`uint32_t gain_1`

Gain 1.

`uint32_t exposure_2`

Exposure time 2.

`uint32_t gain_2`

Gain 2.

## Detailed Description

HDR Configuration.

Definition at line [1422](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ `enable`

```
uint8_t HDR_CONFIG::enable
```

Enable/disable HDR, after enabling HDR, the exposure\_1 and gain\_1 will be used as the first exposure and gain, and the exposure\_2 and gain\_2 will be used as the second exposure and gain. The output image will be alternately exposed and gain between the first and second exposure and gain.

### Attention

After enabling HDR, the auto exposure will be disabled.

Definition at line [1431](#) of file **ObTypes.h**.

#### ◆ sequence\_name

```
uint8_t HDR_CONFIG::sequence_name
```

Sequence name.

Definition at line [1432](#) of file **ObTypes.h**.

#### ◆ exposure\_1

```
uint32_t HDR_CONFIG::exposure_1
```

Exposure time 1.

Definition at line [1433](#) of file **ObTypes.h**.

#### ◆ gain\_1

```
uint32_t HDR_CONFIG::gain_1
```

Gain 1.

Definition at line [1434](#) of file **ObTypes.h**.

#### ◆ exposure\_2

```
uint32_t HDR_CONFIG::exposure_2
```

Exposure time 2.

Definition at line [1435](#) of file [ObTypes.h](#).

◆ [gain\\_2](#)

```
uint32_t HDR_CONFIG::gain_2
```

Gain 2.

Definition at line [1436](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBAccelIntrinsic Member List

This is the complete list of members for **OBAccelIntrinsic**, including all inherited members.

<b>bias</b>	<b>OBAccelIntrinsic</b>
<b>gravity</b>	<b>OBAccelIntrinsic</b>
<b>noiseDensity</b>	<b>OBAccelIntrinsic</b>
<b>randomWalk</b>	<b>OBAccelIntrinsic</b>
<b>referenceTemp</b>	<b>OBAccelIntrinsic</b>
<b>scaleMisalignment</b>	<b>OBAccelIntrinsic</b>
<b>tempSlope</b>	<b>OBAccelIntrinsic</b>

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBAccelIntrinsic Struct Reference

Structure for accelerometer intrinsic parameters. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

double **noiseDensity**

In-run bias instability.

double **randomWalk**

random walk

double **referenceTemp**

reference temperature

double **bias** [3]

bias for x, y, z axis

double **gravity** [3]

gravity direction for x, y, z axis

double **scaleMisalignment** [9]

scale factor and three-axis non-orthogonal error

double **tempSlope** [9]

linear temperature drift coefficient

## Detailed Description

Structure for accelerometer intrinsic parameters.

Definition at line **400** of file [ObTypes.h](#).

## Member Data Documentation

◆ **noiseDensity**

```
double OBAccelIntrinsic::noiseDensity
```

In-run bias instability.

Definition at line [401](#) of file **ObTypes.h**.

◆ **randomWalk**

```
double OBAccelIntrinsic::randomWalk
```

random walk

Definition at line [402](#) of file **ObTypes.h**.

◆ **referenceTemp**

```
double OBAccelIntrinsic::referenceTemp
```

reference temperature

Definition at line [403](#) of file **ObTypes.h**.

◆ **bias**

```
double OBAccelIntrinsic::bias[3]
```

bias for x, y, z axis

Definition at line [404](#) of file **ObTypes.h**.

◆ **gravity**

```
double OBAccelIntrinsic::gravity[3]
```

gravity direction for x, y, z axis

Definition at line [405](#) of file **ObTypes.h**.

◆ **scaleMisalignment**

```
double OBAccelIntrinsic::scaleMisalignment[9]
```

scale factor and three-axis non-orthogonal error

Definition at line [406](#) of file [ObTypes.h](#).

◆ **tempSlope**

```
double OBAccelIntrinsic::tempSlope[9]
```

linear temperature drift coefficient

Definition at line [407](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBAccelValue Member List

This is the complete list of members for **OBAccelValue**, including all inherited members.

**x** [OBAccelValue](#)

**y** [OBAccelValue](#)

**z** [OBAccelValue](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBAccelValue Struct Reference

Data structures for accelerometers and gyroscopes. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

float **x**

X-direction component.

float **y**

Y-direction component.

float **z**

Z-direction component.

## Detailed Description

Data structures for accelerometers and gyroscopes.

Definition at line [638](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **x**

float OBAccelValue::x

X-direction component.

Definition at line [639](#) of file [ObTypes.h](#).

◆ **y**

float OBAccelValue::y

Y-direction component.

Definition at line [640](#) of file [ObTypes.h](#).

◆ Z

float OBAccelValue::z

Z-direction component.

Definition at line **641** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

## OBBoolPropertyRange Member List

This is the complete list of members for **OBBoolPropertyRange**, including all inherited members.

**cur** OBBoolPropertyRange  
**def** OBBoolPropertyRange  
**max** OBBoolPropertyRange  
**min** OBBoolPropertyRange  
**step** OBBoolPropertyRange

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBBoolPropertyRange Struct Reference

Structure for boolean range. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

bool **cur**

Current value.

bool **max**

Maximum value.

bool **min**

Minimum value.

bool **step**

Step value.

bool **def**

Default value.

## Detailed Description

Structure for boolean range.

Definition at line [365](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **cur**

```
bool OBBoolPropertyRange::cur
```

Current value.

Definition at line [366](#) of file [ObTypes.h](#).

◆ **max**

```
bool OBBoolPropertyRange::max
```

Maximum value.

Definition at line [367](#) of file **ObTypes.h**.

◆ **min**

```
bool OBBoolPropertyRange::min
```

Minimum value.

Definition at line [368](#) of file **ObTypes.h**.

◆ **step**

```
bool OBBoolPropertyRange::step
```

Step value.

Definition at line [369](#) of file **ObTypes.h**.

◆ **def**

```
bool OBBoolPropertyRange::def
```

Default value.

Definition at line [370](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBCalibrationParam Member List

This is the complete list of members for **OBCalibrationParam**, including all inherited members.

**distortion** OBCalibrationParam

**extrinsics** OBCalibrationParam

**intrinsics** OBCalibrationParam

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBCalibrationParam Struct Reference

calibration parameters [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

**OBCameraIntrinsic intrinsics [OB\_SENSOR\_TYPE\_COUNT]**

Sensor internal parameters.

**OBCameraDistortion distortion [OB\_SENSOR\_TYPE\_COUNT]**

Sensor distortion.

**OBExtrinsic extrinsics [OB\_SENSOR\_TYPE\_COUNT][OB\_SENSOR\_TYPE\_COUNT]**

## Detailed Description

calibration parameters

Definition at line [467](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ intrinsics

**OBCameraIntrinsic OBCalibrationParam::intrinsics[OB\_SENSOR\_TYPE\_COUNT]**

Sensor internal parameters.

Definition at line [468](#) of file [ObTypes.h](#).

### ◆ distortion

**OBCameraDistortion OBCalibrationParam::distortion[OB\_SENSOR\_TYPE\_COUNT]**

Sensor distortion.

Definition at line [469](#) of file [ObTypes.h](#).

◆ extrinsics

**OBExtrinsic** OBCalibrationParam::extrinsics[**OB\_SENSOR\_TYPE\_COUNT**][**OB\_SENSOR\_TYPE\_COUNT**]

The extrinsic parameters allow 3D coordinate conversions between sensor. To transform from a source to a target 3D coordinate system, under extrinsics[source][target].

Definition at line **470** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBCameraDistortion Member List

This is the complete list of members for **OBCameraDistortion**, including all inherited members.

- k1**    OBCameraDistortion
- k2**    OBCameraDistortion
- k3**    OBCameraDistortion
- k4**    OBCameraDistortion
- k5**    OBCameraDistortion
- k6**    OBCameraDistortion
- model** OBCameraDistortion
- p1**    OBCameraDistortion
- p2**    OBCameraDistortion

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBCameraDistortion Struct Reference

Structure for distortion parameters. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

```
float k1  
    Radial distortion factor 1.  
float k2  
    Radial distortion factor 2.  
float k3  
    Radial distortion factor 3.  
float k4  
    Radial distortion factor 4.  
float k5  
    Radial distortion factor 5.  
float k6  
    Radial distortion factor 6.  
float p1  
    Tangential distortion factor 1.  
float p2  
    Tangential distortion factor 2.
```

**OBCameraDistortionModel model**

## Detailed Description

Structure for distortion parameters.

Definition at line [425](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **k1**

float OBCameraDistortion::k1

Radial distortion factor 1.

Definition at line [426](#) of file **ObTypes.h**.

◆ k2

float OBCameraDistortion::k2

Radial distortion factor 2.

Definition at line [427](#) of file **ObTypes.h**.

◆ k3

float OBCameraDistortion::k3

Radial distortion factor 3.

Definition at line [428](#) of file **ObTypes.h**.

◆ k4

float OBCameraDistortion::k4

Radial distortion factor 4.

Definition at line [429](#) of file **ObTypes.h**.

◆ k5

float OBCameraDistortion::k5

Radial distortion factor 5.

Definition at line [430](#) of file **ObTypes.h**.

◆ k6

```
float OBCameraDistortion::k6
```

Radial distortion factor 6.

Definition at line [431](#) of file **ObTypes.h**.

◆ p1

```
float OBCameraDistortion::p1
```

Tangential distortion factor 1.

Definition at line [432](#) of file **ObTypes.h**.

◆ p2

```
float OBCameraDistortion::p2
```

Tangential distortion factor 2.

Definition at line [433](#) of file **ObTypes.h**.

◆ model

**OBCameraDistortionModel** OBCameraDistortion::model

Definition at line [434](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBCameraIntrinsic Member List

This is the complete list of members for **OBCameraIntrinsic**, including all inherited members.

**cx**    OBCameraIntrinsic

**cy**    OBCameraIntrinsic

**fx**    OBCameraIntrinsic

   OBCameraIntrinsic

**height** OBCameraIntrinsic

**width** OBCameraIntrinsic

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBCameraIntrinsic Struct Reference

Structure for camera intrinsic parameters. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

```
float fx  
    Focal length in x direction.  
float   
    Focal length in y direction.  
float cx  
    Optical center abscissa.  
float cy  
    Optical center ordinate.  
int16_t width  
    Image width.  
int16_t height  
    Image height.
```

## Detailed Description

Structure for camera intrinsic parameters.

Definition at line [388](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ fx

```
float OBCameraIntrinsic::fx
```

Focal length in x direction.

Definition at line [389](#) of file [ObTypes.h](#).

◆ fy

float OBCameraIntrinsic::fy

Focal length in y direction.

Definition at line [390](#) of file **ObTypes.h**.

◆ cx

float OBCameraIntrinsic::cx

Optical center abscissa.

Definition at line [391](#) of file **ObTypes.h**.

◆ cy

float OBCameraIntrinsic::cy

Optical center ordinate.

Definition at line [392](#) of file **ObTypes.h**.

◆ width

int16\_t OBCameraIntrinsic::width

Image width.

Definition at line [393](#) of file **ObTypes.h**.

◆ height

int16\_t OBCameraIntrinsic::height

Image height.

Definition at line [394](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBCameraParam Member List

This is the complete list of members for **OBCameraParam**, including all inherited members.

**depthDistortion** [OBCameraParam](#)

**depthIntrinsic** [OBCameraParam](#)

**isMirrored** [OBCameraParam](#)

**rgbDistortion** [OBCameraParam](#)

**rgbIntrinsic** [OBCameraParam](#)

**transform** [OBCameraParam](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBCameraParam Struct Reference

Structure for camera parameters. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

### **OBCameraIntrinsic depthIntrinsic**

Depth camera internal parameters.

### **OBCameraIntrinsic rgbIntrinsic**

Color camera internal parameters.

### **OBCameraDistortion depthDistortion**

Depth camera distortion parameters.

### **OBCameraDistortion rgbDistortion**

Color camera distortion parameters.

### **OBD2CTransform transform**

Rotation/transformation matrix.

### **bool isMirrored**

Whether the image frame corresponding to this group of parameters is mirrored.

## Detailed Description

Structure for camera parameters.

Definition at line [448](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ **depthIntrinsic**

**OBCameraIntrinsic** OBCameraParam::depthIntrinsic

Depth camera internal parameters.

Definition at line [449](#) of file [ObTypes.h](#).

◆ **rgbIntrinsic**

**OBCameraIntrinsic** OBCameraParam::rgbIntrinsic

Color camera internal parameters.

Definition at line **450** of file **ObTypes.h**.

◆ **depthDistortion**

**OBCameraDistortion** OBCameraParam::depthDistortion

Depth camera distortion parameters.

Definition at line **451** of file **ObTypes.h**.

◆ **rgbDistortion**

**OBCameraDistortion** OBCameraParam::rgbDistortion

Color camera distortion parameters.

Definition at line **452** of file **ObTypes.h**.

◆ **transform**

**OBD2CTransform** OBCameraParam::transform

Rotation/transformation matrix.

Definition at line **453** of file **ObTypes.h**.

◆ **isMirrored**

bool OBCameraParam::isMirrored

Whether the image frame corresponding to this group of parameters is mirrored.

Definition at line **454** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBColorPoint Member List

This is the complete list of members for **OBColorPoint**, including all inherited members.

**b** OBColorPoint

**g** OBColorPoint

**r** OBColorPoint

**x** OBColorPoint

**y** OBColorPoint

**z** OBColorPoint

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## OBColorPoint Struct Reference

3D point structure with color information [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

float **x**

X coordinate.

float **y**

Y coordinate.

float **z**

Z coordinate.

float **r**

Red channel component.

float **g**

Green channel component.

float **b**

Blue channel component.

### Detailed Description

3D point structure with color information

Definition at line [792](#) of file **ObTypes.h**.

### Member Data Documentation

◆ **x**

float OBColorPoint::x

X coordinate.

Definition at line [793](#) of file **ObTypes.h**.

◆ y

float OBColorPoint::y

Y coordinate.

Definition at line [794](#) of file **ObTypes.h**.

◆ z

float OBColorPoint::z

Z coordinate.

Definition at line [795](#) of file **ObTypes.h**.

◆ r

float OBColorPoint::r

Red channel component.

Definition at line [796](#) of file **ObTypes.h**.

◆ g

float OBColorPoint::g

Green channel component.

Definition at line [797](#) of file **ObTypes.h**.

◆ b

float OBColorPoint::b

Blue channel component.

Definition at line [798](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBCompressionParams Member List

This is the complete list of members for **OBCompressionParams**, including all inherited members.

### **threshold** OBCompressionParams

Generated on for OrbtecSDK by  1.14.0

# OBCompressionParams Struct Reference

```
#include <ObTypes.h>
```

## Public Attributes

int **threshold**

## Detailed Description

Compression Params

Definition at line [813](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **threshold**

int OBCompressionParams::threshold

Lossy compression threshold, range [0~255], recommended value is 9, the higher the threshold, the higher the compression ratio.

Definition at line [817](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

## OBD2CTransform Member List

This is the complete list of members for **OBD2CTransform**, including all inherited members.

**rot** **OBD2CTransform**

**trans** **OBD2CTransform**

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

# OBD2CTransform Struct Reference

Structure for rotation/transformation. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

float **rot** [9]

Rotation matrix.

float **trans** [3]

Transformation matrix in millimeters.

## Detailed Description

Structure for rotation/transformation.

Definition at line [440](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **rot**

float OBD2CTransform::rot[9]

Rotation matrix.

Definition at line [441](#) of file [ObTypes.h](#).

◆ **trans**

float OBD2CTransform::trans[3]

Transformation matrix in millimeters.

Definition at line [442](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBDataChunk Member List

This is the complete list of members for **OBDataChunk**, including all inherited members.

<b>data</b>	<b>OBDataChunk</b>
<b>fullDataSize</b>	<b>OBDataChunk</b>
<b>offset</b>	<b>OBDataChunk</b>
<b>size</b>	<b>OBDataChunk</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBDataChunk Struct Reference

Structure for transmitting data blocks. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint8_t * data`

Pointer to current block data.

`uint32_t size`

Length of current block data.

`uint32_t offset`

Offset of current data block relative to complete data.

`uint32_t fullDataSize`

Size of full data.

## Detailed Description

Structure for transmitting data blocks.

Definition at line [311](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ data

`uint8_t* OBDataChunk::data`

Pointer to current block data.

Definition at line [312](#) of file [ObTypes.h](#).

### ◆ size

```
uint32_t OBDataChunk::size
```

Length of current block data.

Definition at line [313](#) of file [\*\*ObTypes.h\*\*](#).

◆ **offset**

```
uint32_t OBDataChunk::offset
```

Offset of current data block relative to complete data.

Definition at line [314](#) of file [\*\*ObTypes.h\*\*](#).

◆ **fullDataSize**

```
uint32_t OBDataChunk::fullDataSize
```

Size of full data.

Definition at line [315](#) of file [\*\*ObTypes.h\*\*](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[\*\*ObTypes.h\*\*](#)

## OBDepthWorkMode Member List

This is the complete list of members for **OBDepthWorkMode**, including all inherited members.

**checksum** OBDepthWorkMode

**name** OBDepthWorkMode

**tag** OBDepthWorkMode

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBDepthWorkMode Struct Reference

Depth work mode. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint8_t checksum [16]`  
Checksum of work mode.

`char name [32]`  
Name of work mode.

### [\*\*OBDepthWorkModeTag tag\*\*](#)

Preset tag.

## Detailed Description

Depth work mode.

Definition at line [978](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ checksum

`uint8_t OBDepthWorkMode::checksum[16]`

Checksum of work mode.

Definition at line [982](#) of file [ObTypes.h](#).

### ◆ name

`char OBDepthWorkMode::name[32]`

Name of work mode.

Definition at line [987](#) of file [ObTypes.h](#).

◆ tag

**OBDepthWorkModeTag** OBDepthWorkMode::tag

Preset tag.

Definition at line **991** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

## OBDeviceSerialNumber Member List

This is the complete list of members for **OBDeviceSerialNumber**, including all inherited members.

**numberStr OBDeviceSerialNumber**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBDeviceSerialNumber Struct Reference

struct of serial number [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

char **numberStr** [16]

## Detailed Description

struct of serial number

Definition at line [1472](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ **numberStr**

char OBDeviceSerialNumber::numberStr[16]

Definition at line [1473](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

## OBDeviceSyncConfig Member List

This is the complete list of members for **OBDeviceSyncConfig**, including all inherited members.

<b>deviceId</b>	<b>OBDeviceSyncConfig</b>
<b>deviceTriggerSignalOutDelay</b>	<b>OBDeviceSyncConfig</b>
<b>deviceTriggerSignalOutPolarity</b>	<b>OBDeviceSyncConfig</b>
<b>irTriggerSignalInDelay</b>	<b>OBDeviceSyncConfig</b>
<b>mcuTriggerFrequency</b>	<b>OBDeviceSyncConfig</b>
<b>rgbTriggerSignalInDelay</b>	<b>OBDeviceSyncConfig</b>
<b>syncMode</b>	<b>OBDeviceSyncConfig</b>

Generated on for OrbbecSDK by **doxygen** 1.14.0

# OBDeviceSyncConfig Struct Reference

Device synchronization configuration. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

### **OBSyncMode syncMode**

Device synchronize mode.

### **uint16\_t irTriggerSignalInDelay**

IR Trigger signal input delay: Used to configure the delay between the IR/Depth/TOF Sensor receiving the trigger signal and starting exposure, Unit: microsecond.

### **uint16\_t rgbTriggerSignalInDelay**

RGB trigger signal input delay is used to configure the delay from the time when an RGB Sensor receives the trigger signal to the time when the exposure starts. Unit: microsecond.

### **uint16\_t deviceTriggerSignalOutDelay**

Device trigger signal output delay, used to control the delay configuration of the host device to output trigger signals or the slave device to output trigger signals. Unit: microsecond.

### **uint16\_t deviceTriggerSignalOutPolarity**

The device trigger signal output polarity is used to control the polarity configuration of the trigger signal output from the host device or the trigger signal output from the slave device.

### **uint16\_t mcuTriggerFrequency**

MCU trigger frequency, used to configure the output frequency of MCU trigger signal in MCU master mode, unit: Hz.

### **uint16\_t deviceId**

Device number. Users can mark the device with this number.

## Detailed Description

Device synchronization configuration.

### **Deprecated**

This structure is deprecated, please use [\*\*ob\\_multi\\_device\\_sync\\_config\*\*](#) instead

Definition at line [912](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ syncMode

**OBSyncMode** OBDeviceSyncConfig::syncMode

Device synchronize mode.

Definition at line **916** of file **ObTypes.h**.

### ◆ irTriggerSignalInDelay

uint16\_t OBDeviceSyncConfig::irTriggerSignalInDelay

IR Trigger signal input delay: Used to configure the delay between the IR/Depth/TOF Sensor receiving the trigger signal and starting exposure, Unit: microsecond.

#### Attention

This parameter is invalid when the synchronization MODE is set to

**OB\_SYNC\_MODE\_PRIMARY\_IR\_TRIGGER**

Definition at line **924** of file **ObTypes.h**.

### ◆ rgbTriggerSignalInDelay

uint16\_t OBDeviceSyncConfig::rgbTriggerSignalInDelay

RGB trigger signal input delay is used to configure the delay from the time when an RGB Sensor receives the trigger signal to the time when the exposure starts. Unit: microsecond.

#### Attention

This parameter is invalid when the synchronization MODE is set to

**OB\_SYNC\_MODE\_PRIMARY**

Definition at line **932** of file **ObTypes.h**.

### ◆ deviceTriggerSignalOutDelay

```
uint16_t OBDeviceSyncConfig::deviceTriggerSignalOutDelay
```

Device trigger signal output delay, used to control the delay configuration of the host device to output trigger signals or the slave device to output trigger signals. Unit: microsecond.

#### Attention

This parameter is invalid when the synchronization MODE is set to  
**OB\_SYNC\_MODE\_CLOSE** or **OB\_SYNC\_MODE\_STANDALONE**

Definition at line **940** of file **ObTypes.h**.

#### ◆ deviceTriggerSignalOutPolarity

```
uint16_t OBDeviceSyncConfig::deviceTriggerSignalOutPolarity
```

The device trigger signal output polarity is used to control the polarity configuration of the trigger signal output from the host device or the trigger signal output from the slave device.

0: forward pulse; 1: negative pulse

#### Attention

This parameter is invalid when the synchronization MODE is set to  
**OB\_SYNC\_MODE\_CLOSE** or **OB\_SYNC\_MODE\_STANDALONE**

Definition at line **949** of file **ObTypes.h**.

#### ◆ mcuTriggerFrequency

```
uint16_t OBDeviceSyncConfig::mcuTriggerFrequency
```

MCU trigger frequency, used to configure the output frequency of MCU trigger signal in MCU master mode, unit: Hz.

This configuration will directly affect the image output frame rate of the Sensor. Unit: FPS (frames per second)

#### Attention

This parameter is invalid only when the synchronization MODE is set to  
**OB\_SYNC\_MODE\_PRIMARY\_MCU\_TRIGGER**

Definition at line **957** of file **ObTypes.h**.

◆ deviceId

uint16\_t OBDeviceSyncConfig::deviceId

Device number. Users can mark the device with this number.

Definition at line [962](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBDeviceTemperature Member List

This is the complete list of members for **OBDeviceTemperature**, including all inherited members.

<b>chipBottomTemp</b>	OBDeviceTemperature
<b>chipTopTemp</b>	OBDeviceTemperature
<b>cpuTemp</b>	OBDeviceTemperature
<b>imuTemp</b>	OBDeviceTemperature
<b>irLeftTemp</b>	OBDeviceTemperature
<b>irRightTemp</b>	OBDeviceTemperature
<b>irTemp</b>	OBDeviceTemperature
<b>ldmTemp</b>	OBDeviceTemperature
<b>mainBoardTemp</b>	OBDeviceTemperature
<b>rgbTemp</b>	OBDeviceTemperature
<b>tecTemp</b>	OBDeviceTemperature

# OBDeviceTemperature Struct Reference

Temperature parameters of the device (unit: Celsius) [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

float **cpuTemp**

CPU temperature.

float **irTemp**

IR temperature.

float **ldmTemp**

Laser temperature.

float **mainBoardTemp**

Motherboard temperature.

float **tecTemp**

TEC temperature.

float **imuTemp**

IMU temperature.

float **rgbTemp**

RGB temperature.

float **irLeftTemp**

Left IR temperature.

float **irRightTemp**

Right IR temperature.

float **chipTopTemp**

MX6600 top temperature.

float **chipBottomTemp**

MX6600 bottom temperature.

## Detailed Description

Temperature parameters of the device (unit: Celsius)

Definition at line **652** of file [ObTypes.h](#).

## Member Data Documentation

### ◆ cpuTemp

float OBDeviceTemperature::cpuTemp

CPU temperature.

Definition at line [653](#) of file [ObTypes.h](#).

### ◆ irTemp

float OBDeviceTemperature::irTemp

IR temperature.

Definition at line [654](#) of file [ObTypes.h](#).

### ◆ ldmTemp

float OBDeviceTemperature::ldmTemp

Laser temperature.

Definition at line [655](#) of file [ObTypes.h](#).

### ◆ mainBoardTemp

float OBDeviceTemperature::mainBoardTemp

Motherboard temperature.

Definition at line [656](#) of file [ObTypes.h](#).

### ◆ tecTemp

```
float OBDeviceTemperature::tecTemp
```

TEC temperature.

Definition at line [657](#) of file **ObTypes.h**.

◆ **imuTemp**

```
float OBDeviceTemperature::imuTemp
```

IMU temperature.

Definition at line [658](#) of file **ObTypes.h**.

◆ **rgbTemp**

```
float OBDeviceTemperature::rgbTemp
```

RGB temperature.

Definition at line [659](#) of file **ObTypes.h**.

◆ **irLeftTemp**

```
float OBDeviceTemperature::irLeftTemp
```

Left IR temperature.

Definition at line [660](#) of file **ObTypes.h**.

◆ **irRightTemp**

```
float OBDeviceTemperature::irRightTemp
```

Right IR temperature.

Definition at line [661](#) of file **ObTypes.h**.

◆ **chipTopTemp**

```
float OBDeviceTemperature::chipTopTemp
```

MX6600 top temperature.

Definition at line **662** of file **ObTypes.h**.

◆ **chipBottomTemp**

```
float OBDeviceTemperature::chipBottomTemp
```

MX6600 bottom temperature.

Definition at line **663** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBDispOffsetConfig Member List

This is the complete list of members for **OBDispOffsetConfig**, including all inherited members.

**enable** OBDispOffsetConfig  
**offset0** OBDispOffsetConfig  
**offset1** OBDispOffsetConfig  
**reserved** OBDispOffsetConfig

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## OBDispOffsetConfig Struct Reference

Disparity offset interleaving configuration. [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

```
uint8_t enable
uint8_t offset0
uint8_t offset1
uint8_t reserved
```

### Detailed Description

Disparity offset interleaving configuration.

Definition at line [1479](#) of file **ObTypes.h**.

### Member Data Documentation

#### ◆ enable

```
uint8_t OBDispOffsetConfig::enable
```

Definition at line [1480](#) of file **ObTypes.h**.

#### ◆ offset0

```
uint8_t OBDispOffsetConfig::offset0
```

Definition at line [1481](#) of file **ObTypes.h**.

#### ◆ offset1

```
uint8_t OBDispOffsetConfig::offset1
```

Definition at line [1482](#) of file [\*\*ObTypes.h\*\*](#).

◆ reserved

```
uint8_t OBDispOffsetConfig::reserved
```

Definition at line [1483](#) of file [\*\*ObTypes.h\*\*](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[\*\*ObTypes.h\*\*](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBDisparityParam Member List

This is the complete list of members for **OBDisparityParam**, including all inherited members.

<b>baseline</b>	<b>OBDisparityParam</b>
<b>bitSize</b>	<b>OBDisparityParam</b>
<b>dispIntPlace</b>	<b>OBDisparityParam</b>
<b>dispOffset</b>	<b>OBDisparityParam</b>
<b>fx</b>	<b>OBDisparityParam</b>
<b>invalidDisp</b>	<b>OBDisparityParam</b>
<b>isDualCamera</b>	<b>OBDisparityParam</b>
<b>minDisparity</b>	<b>OBDisparityParam</b>
<b>packMode</b>	<b>OBDisparityParam</b>
<b>unit</b>	<b>OBDisparityParam</b>
<b>zpd</b>	<b>OBDisparityParam</b>
<b>zpps</b>	<b>OBDisparityParam</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBDisparityParam Struct Reference

disparity parameters for disparity based camera [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

```
double zpd
double zpps
float baseline
double fx
uint8_t bitSize
float unit
float minDisparity
uint8_t packMode
float dispOffset
int32_t invalidDisp
int32_t dispIntPlace
uint8_t isDualCamera
```

## Detailed Description

disparity parameters for disparity based camera

Definition at line [740](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ zpd

```
double OBDisparityParam::zpd
```

Definition at line [741](#) of file [ObTypes.h](#).

### ◆ zpps

double OBDisparityParam::zpps

Definition at line [742](#) of file **ObTypes.h**.

◆ **baseline**

float OBDisparityParam::baseline

Definition at line [743](#) of file **ObTypes.h**.

◆ **fx**

double OBDisparityParam::fx

Definition at line [744](#) of file **ObTypes.h**.

◆ **bitSize**

uint8\_t OBDisparityParam::bitSize

Definition at line [745](#) of file **ObTypes.h**.

◆ **unit**

float OBDisparityParam::unit

Definition at line [746](#) of file **ObTypes.h**.

◆ **minDisparity**

float OBDisparityParam::minDisparity

Definition at line [747](#) of file **ObTypes.h**.

◆ **packMode**

```
uint8_t OBDisparityParam::packMode
```

Definition at line [748](#) of file **ObTypes.h**.

◆ **dispOffset**

```
float OBDisparityParam::dispOffset
```

Definition at line [749](#) of file **ObTypes.h**.

◆ **invalidDisp**

```
int32_t OBDisparityParam::invalidDisp
```

Definition at line [750](#) of file **ObTypes.h**.

◆ **dispIntPlace**

```
int32_t OBDisparityParam::dispIntPlace
```

Definition at line [751](#) of file **ObTypes.h**.

◆ **isDualCamera**

```
uint8_t OBDisparityParam::isDualCamera
```

Definition at line [752](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBEdgeNoiseRemovalFilterParams Member List

This is the complete list of members for **OBEdgeNoiseRemovalFilterParams**, including all inherited members.

**marginBottomTh** OBEdgeNoiseRemovalFilterParams

**marginLeftTh** OBEdgeNoiseRemovalFilterParams

**marginRightTh** OBEdgeNoiseRemovalFilterParams

**marginTopTh** OBEdgeNoiseRemovalFilterParams

**type** OBEdgeNoiseRemovalFilterParams

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## OBEdgeNoiseRemovalFilterParams Struct Reference

```
#include <ObTypes.h>
```

### Public Attributes

#### **OBEdgeNoiseRemovalType type**

```
    uint16_t marginLeftTh  
    uint16_t marginRightTh  
    uint16_t marginTopTh  
    uint16_t marginBottomTh
```

### Detailed Description

Definition at line **1028** of file **ObTypes.h**.

### Member Data Documentation

#### ◆ type

**OBEdgeNoiseRemovalType** OBEdgeNoiseRemovalFilterParams::type

Definition at line **1029** of file **ObTypes.h**.

#### ◆ marginLeftTh

```
uint16_t OBEdgeNoiseRemovalFilterParams::marginLeftTh
```

Definition at line **1030** of file **ObTypes.h**.

#### ◆ marginRightTh

```
uint16_t OBEdgeNoiseRemovalFilterParams::marginRightTh
```

Definition at line **1031** of file **ObTypes.h**.

◆ marginTopTh

```
uint16_t OBEdgeNoiseRemovalFilterParams::marginTopTh
```

Definition at line **1032** of file **ObTypes.h**.

◆ marginBottomTh

```
uint16_t OBEdgeNoiseRemovalFilterParams::marginBottomTh
```

Definition at line **1033** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBFilterConfigSchemaItem Member List

This is the complete list of members for **OBFilterConfigSchemaItem**, including all inherited members.

**def** **OBFilterConfigSchemaItem**  
**desc** **OBFilterConfigSchemaItem**  
**max** **OBFilterConfigSchemaItem**  
**min** **OBFilterConfigSchemaItem**  
**name** **OBFilterConfigSchemaItem**  
**step** **OBFilterConfigSchemaItem**  
**type** **OBFilterConfigSchemaItem**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBFilterConfigSchemaItem Struct Reference

Configuration Item for the filter. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

const char \* **name**

Name of the configuration item.

**OBFilterConfigValueType type**

Value type of the configuration item.

double **min**

Minimum value casted to double.

double **max**

Maximum value casted to double.

double **step**

Step value casted to double.

double **def**

Default value casted to double.

const char \* **desc**

Description of the configuration item.

## Detailed Description

Configuration Item for the filter.

Definition at line **1459** of file **ObTypes.h**.

## Member Data Documentation

◆ **name**

```
const char* OBFilterConfigSchemaItem::name
```

Name of the configuration item.

Definition at line [1460](#) of file [ObTypes.h](#).

◆ type

```
OBFilterConfigValueType OBFilterConfigSchemaItem::type
```

Value type of the configuration item.

Definition at line [1461](#) of file [ObTypes.h](#).

◆ min

```
double OBFilterConfigSchemaItem::min
```

Minimum value casted to double.

Definition at line [1462](#) of file [ObTypes.h](#).

Referenced by [ob::getPropertyRange\(\)](#).

◆ max

```
double OBFilterConfigSchemaItem::max
```

Maximum value casted to double.

Definition at line [1463](#) of file [ObTypes.h](#).

Referenced by [ob::getPropertyRange\(\)](#).

◆ step

```
double OBFilterConfigSchemaItem::step
```

Step value casted to double.

Definition at line **1464** of file **ObTypes.h**.

Referenced by **ob::getPropertyRange()**.

◆ def

```
double OBFilterConfigSchemaItem::def
```

Default value casted to double.

Definition at line **1465** of file **ObTypes.h**.

Referenced by **ob::getPropertyRange()**.

◆ desc

```
const char* OBFilterConfigSchemaItem::desc
```

Description of the configuration item.

Definition at line **1466** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBFloatPropertyRange Member List

This is the complete list of members for **OBFloatPropertyRange**, including all inherited members.

**cur** **OBFloatPropertyRange**  
**def** **OBFloatPropertyRange**  
**max** **OBFloatPropertyRange**  
**min** **OBFloatPropertyRange**  
**step** **OBFloatPropertyRange**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBFloatPropertyRange Struct Reference

Structure for float range. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

float **cur**

Current value.

float **max**

Maximum value.

float **min**

Minimum value.

float **step**

Step value.

float **def**

Default value.

## Detailed Description

Structure for float range.

Definition at line [332](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **cur**

```
float OBFloatPropertyRange::cur
```

Current value.

Definition at line [333](#) of file [ObTypes.h](#).

◆ **max**

```
float OBFloatPropertyRange::max
```

Maximum value.

Definition at line [334](#) of file **ObTypes.h**.

◆ **min**

```
float OBFloatPropertyRange::min
```

Minimum value.

Definition at line [335](#) of file **ObTypes.h**.

◆ **step**

```
float OBFloatPropertyRange::step
```

Step value.

Definition at line [336](#) of file **ObTypes.h**.

◆ **def**

```
float OBFloatPropertyRange::def
```

Default value.

Definition at line [337](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBGyroIntrinsic Member List

This is the complete list of members for **OBGyroIntrinsic**, including all inherited members.

<b>bias</b>	<b>OBGyroIntrinsic</b>
<b>noiseDensity</b>	<b>OBGyroIntrinsic</b>
<b>randomWalk</b>	<b>OBGyroIntrinsic</b>
<b>referenceTemp</b>	<b>OBGyroIntrinsic</b>
<b>scaleMisalignment</b>	<b>OBGyroIntrinsic</b>
<b>tempSlope</b>	<b>OBGyroIntrinsic</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBGyroIntrinsic Struct Reference

Structure for gyroscope intrinsic parameters. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

double **noiseDensity**

In-run bias instability.

double **randomWalk**

random walk

double **referenceTemp**

reference temperature

double **bias** [3]

bias for x, y, z axis

double **scaleMisalignment** [9]

scale factor and three-axis non-orthogonal error

double **tempSlope** [9]

linear temperature drift coefficient

## Detailed Description

Structure for gyroscope intrinsic parameters.

Definition at line [413](#) of file **ObTypes.h**.

## Member Data Documentation

◆ **noiseDensity**

```
double OBGyroIntrinsic::noiseDensity
```

In-run bias instability.

Definition at line [414](#) of file **ObTypes.h**.

◆ randomWalk

double OBGyroIntrinsic::randomWalk

random walk

Definition at line [415](#) of file [ObTypes.h](#).

◆ referenceTemp

double OBGyroIntrinsic::referenceTemp

reference temperature

Definition at line [416](#) of file [ObTypes.h](#).

◆ bias

double OBGyroIntrinsic::bias[3]

bias for x, y, z axis

Definition at line [417](#) of file [ObTypes.h](#).

◆ scaleMisalignment

double OBGyroIntrinsic::scaleMisalignment[9]

scale factor and three-axis non-orthogonal error

Definition at line [418](#) of file [ObTypes.h](#).

◆ tempSlope

double OBGyroIntrinsic::tempSlope[9]

linear temperature drift coefficient

Definition at line [419](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBIntPropertyRange Member List

This is the complete list of members for **OBIntPropertyRange**, including all inherited members.

**cur** **OBIntPropertyRange**  
**def** **OBIntPropertyRange**  
**max** **OBIntPropertyRange**  
**min** **OBIntPropertyRange**  
**step** **OBIntPropertyRange**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBIntPropertyRange Struct Reference

Structure for integer range. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

int32\_t **cur**

Current value.

int32\_t **max**

Maximum value.

int32\_t **min**

Minimum value.

int32\_t **step**

Step value.

int32\_t **def**

Default value.

## Detailed Description

Structure for integer range.

Definition at line [321](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **cur**

```
int32_t OBIntPropertyRange::cur
```

Current value.

Definition at line [322](#) of file [ObTypes.h](#).

◆ **max**

```
int32_t OBIntPropertyRange::max
```

Maximum value.

Definition at line [323](#) of file **ObTypes.h**.

◆ **min**

```
int32_t OBIntPropertyRange::min
```

Minimum value.

Definition at line [324](#) of file **ObTypes.h**.

◆ **step**

```
int32_t OBIntPropertyRange::step
```

Step value.

Definition at line [325](#) of file **ObTypes.h**.

◆ **def**

```
int32_t OBIntPropertyRange::def
```

Default value.

Definition at line [326](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBMGCFilterConfig Member List

This is the complete list of members for **OBMGCFilterConfig**, including all inherited members.

<b>height</b>	<b>OBMGCFilterConfig</b>
<b>limit_x_th</b>	<b>OBMGCFilterConfig</b>
<b>limit_y_th</b>	<b>OBMGCFilterConfig</b>
<b>margin_x_th</b>	<b>OBMGCFilterConfig</b>
<b>margin_y_th</b>	<b>OBMGCFilterConfig</b>
<b>max_radius</b>	<b>OBMGCFilterConfig</b>
<b>max_width_left</b>	<b>OBMGCFilterConfig</b>
<b>max_width_right</b>	<b>OBMGCFilterConfig</b>
<b>width</b>	<b>OBMGCFilterConfig</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBMGCFilterConfig Struct Reference

Configuration for mgc filter. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

```
uint32_t width
uint32_t height
    int max_width_left
    int max_width_right
    int max_radius
    int margin_x_th
    int margin_y_th
    int limit_x_th
    int limit_y_th
```

## Detailed Description

Configuration for mgc filter.

Definition at line [491](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ width

```
uint32_t OBMGCFilterConfig::width
```

Definition at line [492](#) of file [ObTypes.h](#).

### ◆ height

```
uint32_t OBMGCFilterConfig::height
```

Definition at line [493](#) of file [ObTypes.h](#).

◆ **max\_width\_left**

int OBMGCFilterConfig::max\_width\_left

Definition at line [494](#) of file **ObTypes.h**.

◆ **max\_width\_right**

int OBMGCFilterConfig::max\_width\_right

Definition at line [495](#) of file **ObTypes.h**.

◆ **max\_radius**

int OBMGCFilterConfig::max\_radius

Definition at line [496](#) of file **ObTypes.h**.

◆ **margin\_x\_th**

int OBMGCFilterConfig::margin\_x\_th

Definition at line [497](#) of file **ObTypes.h**.

◆ **margin\_y\_th**

int OBMGCFilterConfig::margin\_y\_th

Definition at line [498](#) of file **ObTypes.h**.

◆ **limit\_x\_th**

int OBMGCFilterConfig::limit\_x\_th

Definition at line [499](#) of file **ObTypes.h**.

◆ **limit\_y\_th**

int OBMGCFilterConfig::limit\_y\_th

Definition at line **500** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by **doxygen** 1.14.0

## OBNetIpConfig Member List

This is the complete list of members for **OBNetIpConfig**, including all inherited members.

**address** OBNetIpConfig

**dhcp** OBNetIpConfig

**gateway** OBNetIpConfig

**mask** OBNetIpConfig

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBNetIpConfig Struct Reference

IP address configuration for network devices (IPv4) [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint16_t dhcp`

DHCP status.

`uint8_t address [4]`

IP address (IPv4, big endian: 192.168.1.10, address[0] = 192, address[1] = 168, address[2] = 1, address[3] = 10)

`uint8_t mask [4]`

Subnet mask (big endian)

`uint8_t gateway [4]`

Gateway (big endian)

## Detailed Description

IP address configuration for network devices (IPv4)

Definition at line **1088** of file [ObTypes.h](#).

## Member Data Documentation

◆ `dhcp`

`uint16_t OBNetIpConfig::dhcp`

DHCP status.

### Note

0: static IP; 1: DHCP

Definition at line **1094** of file [ObTypes.h](#).

◆ address

uint8\_t OBNetIpConfig::address[4]

IP address (IPv4, big endian: 192.168.1.10, address[0] = 192, address[1] = 168, address[2] = 1, address[3] = 10)

Definition at line [1099](#) of file [ObTypes.h](#).

◆ mask

uint8\_t OBNetIpConfig::mask[4]

Subnet mask (big endian)

Definition at line [1104](#) of file [ObTypes.h](#).

◆ gateway

uint8\_t OBNetIpConfig::gateway[4]

Gateway (big endian)

Definition at line [1109](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

## OBNoiseRemovalFilterParams Member List

This is the complete list of members for **OBNoiseRemovalFilterParams**, including all inherited members.

**disp\_diff** OBNoiseRemovalFilterParams

**max\_size** OBNoiseRemovalFilterParams

**type** OBNoiseRemovalFilterParams

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## OBNoiseRemovalFilterParams Struct Reference

```
#include <ObTypes.h>
```

### Public Attributes

```
    uint16_t max_size  
    uint16_t disp_diff
```

**OBDDONoiseRemovalType type**

### Detailed Description

Definition at line **1045** of file **ObTypes.h**.

### Member Data Documentation

#### ◆ **max\_size**

```
uint16_t OBNoiseRemovalFilterParams::max_size
```

Definition at line **1046** of file **ObTypes.h**.

Referenced by **ob::NoiseRemovalFilter::getFilterParams()**.

#### ◆ **disp\_diff**

```
uint16_t OBNoiseRemovalFilterParams::disp_diff
```

Definition at line **1047** of file **ObTypes.h**.

Referenced by **ob::NoiseRemovalFilter::getFilterParams()**.

#### ◆ **type**

**OBDDONoiseRemovalType** OBNoiseRemovalFilterParams::type

Definition at line **1048** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by **doxygen** 1.14.0

# OBPoint Member List

This is the complete list of members for **OBPoint**, including all inherited members.

**x** [OBPoint](#)

**y** [OBPoint](#)

**z** [OBPoint](#)

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBPoint Struct Reference

3D point structure in the SDK [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

float **x**

X coordinate.

float **y**

Y coordinate.

float **z**

Z coordinate.

### Detailed Description

3D point structure in the SDK

Definition at line [768](#) of file **ObTypes.h**.

### Member Data Documentation

◆ **x**

float OBPoint::x

X coordinate.

Definition at line [769](#) of file **ObTypes.h**.

◆ **y**

float OBPoint::y

Y coordinate.

Definition at line [770](#) of file **ObTypes.h**.

◆ Z

float OBPoint::z

Z coordinate.

Definition at line [771](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBPoint2f Member List

This is the complete list of members for **OBPoint2f**, including all inherited members.

**x** [OBPoint2f](#)

**y** [OBPoint2f](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

# OBPoint2f Struct Reference

2D point structure in the SDK [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

float **x**

X coordinate.

float **y**

Y coordinate.

## Detailed Description

2D point structure in the SDK

Definition at line [777](#) of file **ObTypes.h**.

## Member Data Documentation

◆ **x**

float OBPoint2f::x

X coordinate.

Definition at line [778](#) of file **ObTypes.h**.

◆ **y**

float OBPoint2f::y

Y coordinate.

Definition at line [779](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBPresetResolutionConfig Member List

This is the complete list of members for **OBPresetResolutionConfig**, including all inherited members.

<b>depthDecimationFactor</b>	<b>OBPresetResolutionConfig</b>
<b>height</b>	<b>OBPresetResolutionConfig</b>
<b>irDecimationFactor</b>	<b>OBPresetResolutionConfig</b>
<b>width</b>	<b>OBPresetResolutionConfig</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## OBPresetResolutionConfig Struct Reference

```
#include <ObTypes.h>
```

### Public Attributes

```
int16_t width  
    width  
int16_t height  
    height  
int irDecimationFactor  
    ir decimation factor  
int depthDecimationFactor  
    depth decimation factor
```

### Detailed Description

Definition at line [457](#) of file [ObTypes.h](#).

### Member Data Documentation

#### ◆ width

```
int16_t OBPresetResolutionConfig::width
```

width

Definition at line [458](#) of file [ObTypes.h](#).

#### ◆ height

```
int16_t OBPresetResolutionConfig::height
```

height

Definition at line [459](#) of file [ObTypes.h](#).

◆ **irDecimationFactor**

int OBPresetResolutionConfig::irDecimationFactor

ir decimation factor

Definition at line **460** of file **ObTypes.h**.

◆ **depthDecimationFactor**

int OBPresetResolutionConfig::depthDecimationFactor

depth decimation factor

Definition at line **461** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBPropertyItem Member List

This is the complete list of members for **OBPropertyItem**, including all inherited members.

<b>id</b>	<b>OBPropertyItem</b>
<b>name</b>	<b>OBPropertyItem</b>
<b>permission</b>	<b>OBPropertyItem</b>
<b>type</b>	<b>OBPropertyItem</b>

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBPropertyItem Struct Reference

Used to describe the characteristics of each property. [More...](#)

```
#include <Property.h>
```

## Public Attributes

```
OBPropertyID id  
const char * name  
OBPropertyType type  
OBPermissionType permission
```

## Detailed Description

Used to describe the characteristics of each property.

Definition at line [893](#) of file [Property.h](#).

## Member Data Documentation

### ◆ id

**OBPropertyID** OBPropertyItem::id

Property ID

Definition at line [894](#) of file [Property.h](#).

### ◆ name

const char\* OBPropertyItem::name

Property name

Definition at line [895](#) of file [Property.h](#).

### ◆ type

**OBPropertyType** OBPropertyItem::type

Property type

Definition at line [896](#) of file **Property.h**.

◆ permission

**OBPermissionType** OBPropertyItem::permission

Property read and write permission

Definition at line [897](#) of file **Property.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**Property.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OBProtocolVersion Member List

This is the complete list of members for **OBProtocolVersion**, including all inherited members.

**major** OBProtocolVersion

**minor** OBProtocolVersion

**patch** OBProtocolVersion

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBProtocolVersion Struct Reference

Control command protocol version number. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint8_t major`

Major version number.

`uint8_t minor`

Minor version number.

`uint8_t patch`

Patch version number.

## Detailed Description

Control command protocol version number.

Definition at line [1054](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **major**

```
uint8_t OBProtocolVersion::major
```

Major version number.

Definition at line [1058](#) of file [ObTypes.h](#).

◆ **minor**

```
uint8_t OBProtocolVersion::minor
```

Minor version number.

Definition at line [1063](#) of file [ObTypes.h](#).

◆ patch

uint8\_t OBProtocolVersion::patch

Patch version number.

Definition at line **1068** of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

## OBRect Member List

This is the complete list of members for **OBRect**, including all inherited members.

**height** OBRect

**width** OBRect

**x** OBRect

**y** OBRect

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBRect Struct Reference

Rectangle. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint32_t x`

Origin coordinate x.

`uint32_t y`

Origin coordinate y.

`uint32_t width`

Rectangle width.

`uint32_t height`

Rectangle height.

## Detailed Description

Rectangle.

Definition at line **525** of file **ObTypes.h**.

## Member Data Documentation

◆ `x`

`uint32_t OBRect::x`

Origin coordinate x.

Definition at line **526** of file **ObTypes.h**.

◆ `y`

```
uint32_t OBRect::y
```

Origin coordinate y.

Definition at line [527](#) of file **ObTypes.h**.

◆ **width**

```
uint32_t OBRect::width
```

Rectangle width.

Definition at line [528](#) of file **ObTypes.h**.

◆ **height**

```
uint32_t OBRect::height
```

Rectangle height.

Definition at line [529](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBSequenceIdItem Member List

This is the complete list of members for **OBSequenceIdItem**, including all inherited members.

**name**            [OBSequenceIdItem](#)

**sequenceSelectId** [OBSequenceIdItem](#)

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

# OBSequenceIdItem Struct Reference

SequenceId fliter list item. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

int **sequenceSelectId**

char **name** [8]

## Detailed Description

SequenceId fliter list item.

Definition at line **998** of file [ObTypes.h](#).

## Member Data Documentation

### ◆ sequenceSelectId

int OBSequenceIdItem::sequenceSelectId

Definition at line **999** of file [ObTypes.h](#).

### ◆ name

char OBSequenceIdItem::name[8]

Definition at line **1000** of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

## OBSpatialAdvancedFilterParams Member List

This is the complete list of members for **OBSpatialAdvancedFilterParams**, including all inherited members.

<b>alpha</b>	<b>OBSpatialAdvancedFilterParams</b>
<b>disp_diff</b>	<b>OBSpatialAdvancedFilterParams</b>
<b>magnitude</b>	<b>OBSpatialAdvancedFilterParams</b>
<b>radius</b>	<b>OBSpatialAdvancedFilterParams</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

# OBSpatialAdvancedFilterParams Struct Reference

```
#include <ObTypes.h>
```

## Public Attributes

uint8\_t **magnitude**

float **alpha**

uint16\_t **disp\_diff**

uint16\_t **radius**

## Detailed Description

Definition at line **1013** of file **ObTypes.h**.

## Member Data Documentation

### ◆ magnitude

uint8\_t OBSPatialAdvancedFilterParams::magnitude

Definition at line **1014** of file **ObTypes.h**.

Referenced by **ob::SpatialAdvancedFilter::getFilterParams()**, and  
**ob::SpatialAdvancedFilter::setFilterParams()**.

### ◆ alpha

float OBSPatialAdvancedFilterParams::alpha

Definition at line **1015** of file **ObTypes.h**.

Referenced by **ob::SpatialAdvancedFilter::getFilterParams()**, and  
**ob::SpatialAdvancedFilter::setFilterParams()**.

### ◆ disp\_diff

```
uint16_t OBSpatialAdvancedFilterParams::disp_diff
```

Definition at line [1016](#) of file [ObTypes.h](#).

Referenced by [ob::SpatialAdvancedFilter::getFilterParams\(\)](#), and  
[ob::SpatialAdvancedFilter::setFilterParams\(\)](#).

◆ **radius**

```
uint16_t OBSpatialAdvancedFilterParams::radius
```

Definition at line [1017](#) of file [ObTypes.h](#).

Referenced by [ob::SpatialAdvancedFilter::getFilterParams\(\)](#), and  
[ob::SpatialAdvancedFilter::setFilterParams\(\)](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

## OBTofExposureThresholdControl Member List

This is the complete list of members for **OBTofExposureThresholdControl**, including all inherited members.

**lower** OBTofExposureThresholdControl

**upper** OBTofExposureThresholdControl

Generated on for OrbbeeSDK by [doxygen](#) 1.14.0

# OBTofExposureThresholdControl Struct Reference

TOF Exposure Threshold. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

int32\_t **upper**  
Upper threshold, unit: ms.

int32\_t **lower**  
Lower threshold, unit: ms.

## Detailed Description

TOF Exposure Threshold.

Definition at line [823](#) of file [ObTypes.h](#).

## Member Data Documentation

### ◆ upper

int32\_t OBTofExposureThresholdControl::upper

Upper threshold, unit: ms.

Definition at line [824](#) of file [ObTypes.h](#).

### ◆ lower

int32\_t OBTofExposureThresholdControl::lower

Lower threshold, unit: ms.

Definition at line [825](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## OB UInt16PropertyRange Member List

This is the complete list of members for **OB UInt16PropertyRange**, including all inherited members.

**cur** OB UInt16PropertyRange  
**def** OB UInt16PropertyRange  
**max** OB UInt16PropertyRange  
**min** OB UInt16PropertyRange  
**step** OB UInt16PropertyRange

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBUInt16PropertyRange Struct Reference

Structure for float range. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

`uint16_t cur`

Current value.

`uint16_t max`

Maximum value.

`uint16_t min`

Minimum value.

`uint16_t step`

Step value.

`uint16_t def`

Default value.

## Detailed Description

Structure for float range.

Definition at line [343](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ `cur`

`uint16_t OBUInt16PropertyRange::cur`

Current value.

Definition at line [344](#) of file [ObTypes.h](#).

◆ `max`

```
uint16_t OB UInt16PropertyRange::max
```

Maximum value.

Definition at line [345](#) of file **ObTypes.h**.

◆ **min**

```
uint16_t OB UInt16PropertyRange::min
```

Minimum value.

Definition at line [346](#) of file **ObTypes.h**.

◆ **step**

```
uint16_t OB UInt16PropertyRange::step
```

Step value.

Definition at line [347](#) of file **ObTypes.h**.

◆ **def**

```
uint16_t OB UInt16PropertyRange::def
```

Default value.

Definition at line [348](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OB UInt8PropertyRange Member List

This is the complete list of members for **OB UInt8PropertyRange**, including all inherited members.

**cur** OB UInt8PropertyRange  
**def** OB UInt8PropertyRange  
**max** OB UInt8PropertyRange  
**min** OB UInt8PropertyRange  
**step** OB UInt8PropertyRange

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBUInt8PropertyRange Struct Reference

Structure for float range. [More...](#)

```
#include <ObTypes.h>
```

## Public Attributes

uint8\_t **cur**

Current value.

uint8\_t **max**

Maximum value.

uint8\_t **min**

Minimum value.

uint8\_t **step**

Step value.

uint8\_t **def**

Default value.

## Detailed Description

Structure for float range.

Definition at line [354](#) of file [ObTypes.h](#).

## Member Data Documentation

◆ **cur**

```
uint8_t OBUInt8PropertyRange::cur
```

Current value.

Definition at line [355](#) of file [ObTypes.h](#).

◆ **max**

```
uint8_t OB UInt8PropertyRange::max
```

Maximum value.

Definition at line [356](#) of file **ObTypes.h**.

◆ **min**

```
uint8_t OB UInt8PropertyRange::min
```

Minimum value.

Definition at line [357](#) of file **ObTypes.h**.

◆ **step**

```
uint8_t OB UInt8PropertyRange::step
```

Step value.

Definition at line [358](#) of file **ObTypes.h**.

◆ **def**

```
uint8_t OB UInt8PropertyRange::def
```

Default value.

Definition at line [359](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## OBXYTables Member List

This is the complete list of members for **OBXYTables**, including all inherited members.

**height** OBXYTables

**width** OBXYTables

**xTable** OBXYTables

**yTable** OBXYTables

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

# OBXYTables Struct Reference

```
#include <ObTypes.h>
```

## Public Attributes

```
float * xTable  
    table used to compute X coordinate  
float * yTable  
    table used to compute Y coordinate  
int width  
    width of x and y tables  
int height  
    height of x and y tables
```

## Detailed Description

Definition at line [782](#) of file **ObTypes.h**.

## Member Data Documentation

### ◆ xTable

```
float* OBXYTables::xTable
```

table used to compute X coordinate

Definition at line [783](#) of file **ObTypes.h**.

### ◆ yTable

```
float* OBXYTables::yTable
```

table used to compute Y coordinate

Definition at line [784](#) of file **ObTypes.h**.

◆ width

int OBXYTables::width

width of x and y tables

Definition at line [785](#) of file **ObTypes.h**.

◆ height

int OBXYTables::height

height of x and y tables

Definition at line [786](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob::RangeTraits< T > Member List

This is the complete list of members for **ob::RangeTraits< T >**, including all inherited members.

**valueType** typedef **ob::RangeTraits< T >**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob::RangeTraits< T > Struct Template Reference

Get the type of a PropertyRange member. [More...](#)

```
#include <Filter.hpp>
```

### Public Types

```
using valueType = void
```

### Detailed Description

```
template<typename T>
struct ob::RangeTraits< T >
```

Get the type of a PropertyRange member.

Definition at line [37](#) of file [Filter.hpp](#).

### Member Typedef Documentation

#### ◆ valueType

```
template<typename T>
using ob::RangeTraits< T >::valueType = void
```

Definition at line [38](#) of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::RangeTraits< OBFloatPropertyRange > Member List

This is the complete list of members for **ob::RangeTraits< OBFloatPropertyRange >**, including all inherited members.

**valueType** typedef **ob::RangeTraits< OBFloatPropertyRange >**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob::RangeTraits< OBFloatPropertyRange > Struct Reference

```
#include <Filter.hpp>
```

### Public Types

```
using valueType = float
```

### Detailed Description

Definition at line [53](#) of file [Filter.hpp](#).

### Member Typedef Documentation

#### ◆ valueType

```
using ob::RangeTraits< OBFloatPropertyRange >::valueType = float
```

Definition at line [54](#) of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::RangeTraits< OBIntPropertyRange > Member List

This is the complete list of members for **ob::RangeTraits< OBIntPropertyRange >**, including all inherited members.

**valueType** typedef **ob::RangeTraits< OBIntPropertyRange >**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob::RangeTraits< OBIntPropertyRange > Struct Reference

```
#include <Filter.hpp>
```

### Public Types

```
using valueType = uint32_t
```

### Detailed Description

Definition at line **49** of file [Filter.hpp](#).

### Member Typedef Documentation

#### ◆ valueType

```
using ob::RangeTraits< OBIntPropertyRange >::valueType = uint32_t
```

Definition at line **50** of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::RangeTraits< OB UInt16PropertyRange > Member List

This is the complete list of members for **ob::RangeTraits< OB UInt16PropertyRange >**, including all inherited members.

**valueType** typedef **ob::RangeTraits< OB UInt16PropertyRange >**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob::RangeTraits< OB UInt16PropertyRange > Struct Reference

```
#include <Filter.hpp>
```

### Public Types

```
using valueType = uint16_t
```

### Detailed Description

Definition at line **45** of file [Filter.hpp](#).

### Member Typedef Documentation

#### ◆ valueType

```
using ob::RangeTraits< OB UInt16PropertyRange >::valueType = uint16_t
```

Definition at line **46** of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob::RangeTraits< OB UInt8PropertyRange > Member List

This is the complete list of members for **ob::RangeTraits< OB UInt8PropertyRange >**, including all inherited members.

**valueType** typedef **ob::RangeTraits< OB UInt8PropertyRange >**

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob::RangeTraits< OB UInt8PropertyRange > Struct Reference

```
#include <Filter.hpp>
```

### Public Types

```
using valueType = uint8_t
```

### Detailed Description

Definition at line **41** of file [Filter.hpp](#).

### Member Typedef Documentation

#### ◆ valueType

```
using ob::RangeTraits< OB UInt8PropertyRange >::valueType = uint8_t
```

Definition at line **42** of file [Filter.hpp](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor.hpp/[Filter.hpp](#)

## ob\_device\_timestamp\_reset\_config Member List

This is the complete list of members for **ob\_device\_timestamp\_reset\_config**, including all inherited members.

<b>enable</b>	<a href="#">ob_device_timestamp_reset_config</a>
<b>timestamp_reset_delay_us</b>	<a href="#">ob_device_timestamp_reset_config</a>
<b>timestamp_reset_signal_output_enable</b>	<a href="#">ob_device_timestamp_reset_config</a>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob\_device\_timestamp\_reset\_config Struct Reference

The timestamp reset configuration of the device. [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

bool **enable**

Whether to enable the timestamp reset function.

int **timestamp\_reset\_delay\_us**

The delay time of executing the timestamp reset function after receiving the command or signal in microseconds.

bool **timestamp\_reset\_signal\_output\_enable**

the timestamp reset signal output enable flag.

### Detailed Description

The timestamp reset configuration of the device.

Definition at line [1381](#) of file [ObTypes.h](#).

### Member Data Documentation

◆ **enable**

```
bool ob_device_timestamp_reset_config::enable
```

Whether to enable the timestamp reset function.

If the timestamp reset function is enabled, the timer for calculating the timestamp for output frames will be reset to 0 when the timestamp reset command or timestamp reset signal is received, and one timestamp reset signal will be output via TIMER\_SYNC\_OUT pin on synchronization port by default. The timestamp reset signal is input via TIMER\_SYNC\_IN pin on the synchronization port.

#### Attention

For some models, the timestamp reset function is always enabled and cannot be disabled.

Definition at line [1390](#) of file **ObTypes.h**.

#### ◆ timestamp\_reset\_delay\_us

```
int ob_device_timestamp_reset_config::timestamp_reset_delay_us
```

The delay time of executing the timestamp reset function after receiving the command or signal in microseconds.

Definition at line [1395](#) of file **ObTypes.h**.

#### ◆ timestamp\_reset\_signal\_output\_enable

```
bool ob_device_timestamp_reset_config::timestamp_reset_signal_output_enable
```

the timestamp reset signal output enable flag.

#### Attention

For some models, the timestamp reset signal output is always enabled and cannot be disabled.

Definition at line [1402](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## ob\_error Member List

This is the complete list of members for **ob\_error**, including all inherited members.

<b>args</b>	<b>ob_error</b>
<b>exception_type</b>	<b>ob_error</b>
<b>function</b>	<b>ob_error</b>
<b>message</b>	<b>ob_error</b>
<b>status</b>	<b>ob_error</b>

Generated on for OrbtecSDK by [doxygen](#) 1.14.0

## ob\_error Struct Reference

The error class exposed by the SDK, users can get detailed error information according to the error. [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

#### **ob\_status status**

Describe the status code of the error, as compatible with previous customer status code requirements.

char **message** [256]

Describe the detailed error log.

char **function** [256]

Describe the name of the function where the error occurred.

char **args** [256]

Describes the parameters passed to the function when an error occurs. Used to check whether the parameter is wrong.

#### **ob\_exception\_type exception\_type**

The description is the specific error type of the SDK.

### Detailed Description

The error class exposed by the SDK, users can get detailed error information according to the error.

Definition at line [117](#) of file **ObTypes.h**.

### Member Data Documentation

#### ◆ **status**

**ob\_status** ob\_error::status

Describe the status code of the error, as compatible with previous customer status code requirements.

Definition at line [118](#) of file **ObTypes.h**.

#### ◆ **message**

```
char ob_error::message[256]
```

Describe the detailed error log.

Definition at line [119](#) of file **ObTypes.h**.

◆ **function**

```
char ob_error::function[256]
```

Describe the name of the function where the error occurred.

Definition at line [120](#) of file **ObTypes.h**.

◆ **args**

```
char ob_error::args[256]
```

Describes the parameters passed to the function when an error occurs. Used to check whether the parameter is wrong.

Definition at line [121](#) of file **ObTypes.h**.

◆ **exception\_type**

**ob\_exception\_type** ob\_error::exception\_type

The description is the specific error type of the SDK.

Definition at line [122](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**

## ob\_margin\_filter\_config Member List

This is the complete list of members for **ob\_margin\_filter\_config**, including all inherited members.

<b>enable_direction</b>	ob_margin_filter_config
<b>height</b>	ob_margin_filter_config
<b>limit_x_th</b>	ob_margin_filter_config
<b>limit_y_th</b>	ob_margin_filter_config
<b>margin_x_th</b>	ob_margin_filter_config
<b>margin_y_th</b>	ob_margin_filter_config
<b>width</b>	ob_margin_filter_config

Generated on for OrbbecSDK by [doxygen](#) 1.14.0

## ob\_margin\_filter\_config Struct Reference

Configuration for depth margin filter. [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

int **margin\_x\_th**

Horizontal threshold settings.

int **margin\_y\_th**

Vertical threshold settings.

int **limit\_x\_th**

Maximum horizontal threshold.

int **limit\_y\_th**

Maximum vertical threshold.

uint32\_t **width**

Image width.

uint32\_t **height**

Image height.

bool **enable\_direction**

Set to true for horizontal and vertical, false for horizontal only.

### Detailed Description

Configuration for depth margin filter.

Definition at line [478](#) of file [ObTypes.h](#).

### Member Data Documentation

◆ **margin\_x\_th**

```
int ob_margin_filter_config::margin_x_th
```

Horizontal threshold settings.

Definition at line [479](#) of file **ObTypes.h**.

◆ **margin\_y\_th**

```
int ob_margin_filter_config::margin_y_th
```

Vertical threshold settings.

Definition at line [480](#) of file **ObTypes.h**.

◆ **limit\_x\_th**

```
int ob_margin_filter_config::limit_x_th
```

Maximum horizontal threshold.

Definition at line [481](#) of file **ObTypes.h**.

◆ **limit\_y\_th**

```
int ob_margin_filter_config::limit_y_th
```

Maximum vertical threshold.

Definition at line [482](#) of file **ObTypes.h**.

◆ **width**

```
uint32_t ob_margin_filter_config::width
```

Image width.

Definition at line [483](#) of file **ObTypes.h**.

◆ **height**

```
uint32_t ob_margin_filter_config::height
```

Image height.

Definition at line [484](#) of file [ObTypes.h](#).

◆ [enable\\_direction](#)

```
bool ob_margin_filter_config::enable_direction
```

Set to true for horizontal and vertical, false for horizontal only.

Definition at line [485](#) of file [ObTypes.h](#).

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/[ObTypes.h](#)

## ob\_multi\_device\_sync\_config Member List

This is the complete list of members for **ob\_multi\_device\_sync\_config**, including all inherited members.

<b>colorDelayUs</b>	<code>ob_multi_device_sync_config</code>
<b>depthDelayUs</b>	<code>ob_multi_device_sync_config</code>
<b>framesPerTrigger</b>	<code>ob_multi_device_sync_config</code>
<b>syncMode</b>	<code>ob_multi_device_sync_config</code>
<b>trigger2ImageDelayUs</b>	<code>ob_multi_device_sync_config</code>
<b>triggerOutDelayUs</b>	<code>ob_multi_device_sync_config</code>
<b>triggerOutEnable</b>	<code>ob_multi_device_sync_config</code>

## ob\_multi\_device\_sync\_config Struct Reference

The synchronization configuration of the device. [More...](#)

```
#include <ObTypes.h>
```

### Public Attributes

#### **OBMultiDeviceSyncMode syncMode**

The sync mode of the device.

##### int **depthDelayUs**

The delay time of the depth image capture after receiving the capture command or trigger signal in microseconds.

##### int **colorDelayUs**

The delay time of the color image capture after receiving the capture command or trigger signal in microseconds.

##### int **trigger2ImageDelayUs**

The delay time of the image capture after receiving the capture command or trigger signal in microseconds.

##### bool **triggerOutEnable**

Trigger signal output enable flag.

##### int **triggerOutDelayUs**

The delay time of the trigger signal output after receiving the capture command or trigger signal in microseconds.

##### int **framesPerTrigger**

The frame number of each stream after each trigger in triggering mode.

### Detailed Description

The synchronization configuration of the device.

Definition at line [1317](#) of file [ObTypes.h](#).

### Member Data Documentation

#### ◆ syncMode

## **OBMultiDeviceSyncMode** ob\_multi\_device\_sync\_config::syncMode

The sync mode of the device.

Definition at line **1321** of file **ObTypes.h**.

### ◆ depthDelayUs

```
int ob_multi_device_sync_config::depthDelayUs
```

The delay time of the depth image capture after receiving the capture command or trigger signal in microseconds.

#### **Attention**

This parameter is only valid for some models, please refer to the product manual for details.

Definition at line **1328** of file **ObTypes.h**.

### ◆ colorDelayUs

```
int ob_multi_device_sync_config::colorDelayUs
```

The delay time of the color image capture after receiving the capture command or trigger signal in microseconds.

#### **Attention**

This parameter is only valid for some models, please refer to the product manual for details.

Definition at line **1335** of file **ObTypes.h**.

### ◆ trigger2ImageDelayUs

```
int ob_multi_device_sync_config::trigger2ImageDelayUs
```

The delay time of the image capture after receiving the capture command or trigger signal in microseconds.

The depth and color images are captured synchronously as the product design and can not change the delay between the depth and color images.

### Attention

For Orbtec Astra 2 device, this parameter is valid only when the **triggerOutDelayUs** is set to 0.

This parameter is only valid for some models to replace **depthDelayUs** and **colorDelayUs**, please refer to the product manual for details.

Definition at line **1345** of file **ObTypes.h**.

### ◆ triggerOutEnable

```
bool ob_multi_device_sync_config::triggerOutEnable
```

Trigger signal output enable flag.

After the trigger signal output is enabled, the trigger signal will be output when the capture command or trigger signal is received. User can adjust the delay time of the trigger signal output by **triggerOutDelayUs**.

### Attention

For some models, the trigger signal output is always enabled and cannot be disabled.

If device is in the **OB\_MULTI\_DEVICE\_SYNC\_MODE\_FREE\_RUN** or **OB\_MULTI\_DEVICE\_SYNC\_MODE\_STANDALONE** mode, the trigger signal output is always disabled. Set this parameter to true will not take effect.

Definition at line **1356** of file **ObTypes.h**.

### ◆ triggerOutDelayUs

```
int ob_multi_device_sync_config::triggerOutDelayUs
```

The delay time of the trigger signal output after receiving the capture command or trigger signal in microseconds.

#### Attention

For Orbbec Astra 2 device, only supported -1 and 0. -1 means the trigger signal output delay is automatically adjusted by the device, 0 means the trigger signal output is disabled.

Definition at line [1364](#) of file **ObTypes.h**.

#### ◆ framesPerTrigger

```
int ob_multi_device_sync_config::framesPerTrigger
```

The frame number of each stream after each trigger in triggering mode.

#### Attention

This parameter is only valid when the triggering mode is set to

**OB\_MULTI\_DEVICE\_SYNC\_MODE\_HARDWARE\_TRIGGERING** or  
**OB\_MULTI\_DEVICE\_SYNC\_MODE\_SOFTWARE\_TRIGGERING**.

The trigger frequency multiplied by the number of frames per trigger cannot exceed the maximum frame rate of the stream profile which is set when starting the stream.

Definition at line [1374](#) of file **ObTypes.h**.

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

- E:/code/orbbsdk\_group/OpenOrbbecSDK/include/libobsensor/h/**ObTypes.h**