

Geko

Generated by Doxygen 1.8.6

Thu Dec 24 2015 17:13:04

## Contents

|          |                                  |           |
|----------|----------------------------------|-----------|
| <b>1</b> | <b>Module Documentation</b>      | <b>1</b>  |
| 1.1      | MagEngineAPI                     | 1         |
| 1.1.1    | Detailed Description             | 4         |
| 1.1.2    | Typedef Documentation            | 4         |
| 1.1.3    | Enumeration Type Documentation   | 4         |
| 1.1.4    | Function Documentation           | 7         |
| 1.2      | MediaSourceBase                  | 12        |
| 1.2.1    | Detailed Description             | 12        |
| <b>2</b> | <b>Class Documentation</b>       | <b>13</b> |
| 2.1      | EventValue Struct Reference      | 13        |
| 2.1.1    | Detailed Description             | 14        |
| 2.1.2    | Member Enumeration Documentation | 14        |
| 2.1.3    | Member Data Documentation        | 14        |
| 2.2      | FilterParameter Struct Reference | 14        |
| 2.2.1    | Detailed Description             | 15        |
| 2.3      | GekoWriterState Struct Reference | 15        |
| 2.3.1    | Detailed Description             | 15        |
| 2.3.2    | Member Data Documentation        | 15        |
| 2.4      | MediaSourceBase Class Reference  | 16        |
| 2.4.1    | Member Function Documentation    | 16        |
|          | <b>Index</b>                     | <b>19</b> |

## 1 Module Documentation

### 1.1 MagEngineAPI

#### Classes

- struct [EventValue](#)
- struct [FilterParameter](#)
- struct [GekoWriterState](#)

#### Macros

- #define [UNAVAILABLE](#) `std::numeric_limits<int64_t>::min()`  
*minimum number of int64, used to indicate the video source access is unavailable*
- #define [NULLQUEUE](#) `-1`  
*error code for not set the maximum size of signal data queue*
- #define [DEFAULT\\_ALPHA](#) `40`  
*Default alpha.*
- #define [DEFAULT\\_LOW\\_CUTOFF](#) `0.4`

- Default low cutoff of the temporal filter.*
- #define `DEFAULT_HIGH_CUTOFF` 2
  - Default high cutoff of the temporal filter.*
- #define `DEFAULT_LAMBDA` 72
  - Default spatial frequency cutoff.*
- #define `DEFAULT_LEVEL` 4
  - Default level of Gaussian pyramid.*
- #define `DEFAULT_RADIAL` 2
  - Default number of octave in a spatial bands (used in steerable motion only)*
- #define `DEFAULT_ANGULAR` 2
  - Default number of orientation in steerable pyramid (used in steerable motion only)*
- #define `DEFAULT_SIGMA` 5
  - Default standard deviation (used in steerable motion only)*
- #define `DEFAULT_CUTOFF_BOUND` 20
  - Default filter cutoff maximum.*
- #define `DEFAULT_LAMBDA_BOUND` 100
  - Default spatial frequency maximum.*

### Typedefs

- typedef void(\* `CallbackFunction` )(struct `EventValue`)

### Enumerations

- enum `MagEngineEvent` {  
`NONE_EVENT = -1`, `ORIGINAL_IMAGE = 0`, `PROCESSED_IMAGE`, `FACE_IMAGE`,  
`NO_FACE_IMAGE`, `FREQUENCY_NUMBER`, `SIGNAL_DATA`, `ROI_RESET`,  
`ROI_CHANGED`, `FILTER_CHANGED`, `VALUE_ALPHA`, `VALUE_LOWCUT`,  
`VALUE_HIGHCUT`, `VALUE_LAMBDA`, `VALUE_LEVEL`, `VALUE_RADIAL`,  
`VALUE_ANGULAR`, `VALUE_SIGMA`, `VALUE_SOURCE_FRAME_RATE`, `VALUE_PROCESS_FRAME_RA-`  
`TE`,  
`VALUE_FILE_DURATION`, `VALUE_FILE_PROGRESS`, `VALUE_CAMERA_COUNT`, `AUTOFOCUS_AVAIL-`  
`ABLE`,  
`AUTOFOCUS_ENABLED`, `OPEN_CAMERA_SUCCESS`, `OPEN_CAMERA_FAIL`, `OPEN_FILE_SUCCESS`,  
`OPEN_FILE_FAIL`, `OPEN_EXTERNAL_SOURCE_SUCCESS`, `OPEN_EXTERNAL_SOURCE_FAIL`, `CLOS-`  
`E_SOURCE_SUCCESS`,  
`FILE_ARCHIVED`, `END_OF_FILE` }
- enum `ProcessMethod` { `LINEAR_MOTION = 0`, `LINEAR_COLOR`, `STEERABLE_MOTION`, `RIESZ_MOTION` }
- enum `SourceOperation` { `CLOSE_SOURCE`, `OPEN_CAMERA_SOURCE`, `OPEN_FILE_SOURCE`, `OPEN_-`  
`EXTERNAL_SOURCE` }
- enum `RoiMode` { `MANUAL_ROI = 0`, `FACE_DETECT`, `FACE_TRACK`, `MARKER_DETECT` }
- enum `SourceAccess` {  
`CAMERA_EXPOSURE = 0`, `CAMERA_AUTO_FOCUS`, `CAMERA_FRAME_RATE`, `CAMERA_BRIGHTNES-`  
`S`,  
`CAMERA_CONTRAST`, `CAMERA_SATURATION`, `CAMERA_AUTO_WHITEBALANCE`, `CAMERA_BACKL-`  
`IGHT_COMPENSATION`,  
`CAMERA_ABSOLUTE_FOCUS`, `CAMERA_ZOOM_IN`, `CAMERA_ZOOM_OUT`, `CAMERA_ZOOM_RESET`,  
`CAMERA_PAN_LEFT`, `CAMERA_PAN_RIGHT`, `CAMERA_PAN_RESET`, `CAMERA_TILT_UP`,  
`CAMERA_TILT_DOWN`, `CAMERA_TILT_RESET`, `CAMERA_PTZ_RESET`, `FILE_SEEK = 100`,  
`FILE_PROGRESS`, `FILE_DURATION`, `FILE_PLAY`, `FILE_PAUSE`,  
`FILE_FORWARD`, `FILE_BACKWARD`, `FILE_STOP`, `FILE_NORNAL_PLAYBACK`,  
`FILE_TURBO_PLAYBACK`, `IMAGE_FLIP = 200` }

- enum `ParameterType` {  
`ALPHA = 0, HIGHCUT, LOWCUT, LEVEL,`  
`LEVEL_BOUND, LAMBDA, LAMBDA_BOUND, RADIAL_OCTAVE,`  
`ANGULAR_ORDER, SIGMA, FRAMERATE, CUTOFF_BOUND` }
- enum `LicenseResponse` {  
`LICENSE_OFFLINE_WARNING = -2, LICENSE_OFFLINE_SUCCESS = -1, LICENSE_SUCCESS = 0, LIC-`  
`ENSE_OFFLINE_FORBIDDEN,`  
`LICENSE_MISSING, LICENSE_INCOMPLETE, LICENSE_EXPIRED` }
- enum `RecordType` { `RECORD_NONE = 0, RECORD_ORIGINAL_ONLY, RECORD_PROCESSED_ONLY,`  
`RECORD_BOTH` }

## Functions

- enum `LicenseResponse` `initMagEngine` (`CallbackFunction` cb)  
*Initialize function.*
- void `destroyMagEngine` ()  
*Release function.*
- void `setProcessMethod` (enum `ProcessMethod` processMethod)  
*Algorithm setting function.*
- void `setSource` (enum `SourceOperation` operation, const char \*source)  
*Video source setting.*
- void `setOutput` (const char \*output)  
*Output filename setting.*
- void `setWebBackendMode` (bool webBackendFlag)  
*Toggle web backend mode.*
- bool `controlSource` (enum `SourceAccess` type, int value=0)  
*Camera or file control.*
- int `getCameraCount` ()  
*Get the number of connected cameras.*
- int64\_t `querySource` (enum `SourceAccess` type)  
*Camera or file query.*
- void `setROIMode` (enum `RoiMode` mode)  
*Region of interest method setting.*
- enum `RoiMode` `getROIMode` ()  
*Get the current ROI mode.*
- void `setROIView` (uint16\_t x, uint16\_t y, uint16\_t width, uint16\_t height)  
*Set Region of interest.*
- void `setFilterParameter` (enum `ParameterType` type, float value)  
*Filter parameter setting.*
- void `setFaceDectionXML` (std::string xmlFile)  
*Face detection XML selection.*
- std::string `getFaceDectionXML` ()  
*Get the name of the selected XML file.*
- void `setFileOutput` (enum `RecordType` recordType)  
*Enable or disable file output.*
- enum `ProcessMethod` `getProcessMethod` ()  
*Get the current process method.*
- std::string `getSourceName` ()  
*Get the video source name.*
- struct `FilterParameter` `getFilterParameter` ()  
*Get the FilterParameter structure.*
- struct `GekoWriterState` `getFileOutputState` ()

*Get the current info about writer.*

- void [setDataQueueLength](#) (size\_t len)  
*set the maximum data queue length for rate calculation*
- size\_t [getDataQueueLength](#) ()  
*get the currently maximum data queue length*
- time\_t [getExpiredDate](#) ()
- int [getVerificationErrorCode](#) ()
- std::string [MAG\\_ENGINE\\_VERSION](#) ()
- const char \* [MAG\\_ENGINE\\_VERSION\\_DATE](#) ()

### 1.1.1 Detailed Description

#### Author

Nathan Chen

#### Date

23 Sep 2015

The API header file, define API functions, callback event and operations

### 1.1.2 Typedef Documentation

#### 1.1.2.1 typedef void(\* CallBackFunction)(struct EventValue)

Callback function

### 1.1.3 Enumeration Type Documentation

#### 1.1.3.1 enum MagEngineEvent

Here is the list of event that will be evoked in Geko, to catch the value and image you are interested in, implement the event handler like we did in xxx.cpp

#### See Also

xxx.cpp

#### Enumerator

**NONE\_EVENT** none event

**ORIGINAL\_IMAGE** Evoked when source image is captured

**PROCESSED\_IMAGE** Evoked when processed image is done

**FACE\_IMAGE** Evoked when face detection success, return the captured face image

**NO\_FACE\_IMAGE** Evoked when face detection fail, return a "no face" image

**FREQUENCY\_NUMBER** Evoked when frequency result had been estimated, NOTE: won't evoked in default, see setDataQueueLength

**SIGNAL\_DATA** Evoked when signal extracted, return the signal queue, NOTE: won't evoked in default, see setDataQueueLength

**ROI\_RESET** Evoked when region of interest had been changed

**ROI\_CHANGED** Evoked when ROI MODE had been changed

**FILTER\_CHANGED** Evoked when filter method had been changed

**VALUE\_ALPHA** Evoked when magnification value had been changed

**VALUE\_LOWCUT** Evoked when filter low cutoff had been changed

**VALUE\_HIGHCUT** Evoked when filter high cutoff had been changed

**VALUE\_LAMBDA** Evoked when spatial frequency cutoff had been changed (used in linear motion only)

**VALUE\_LEVEL** Evoked when Gaussian pyramid level had been changed (used in linear color only)

**VALUE\_RADIAL** Evoked when radial octave had been changed (used in steerable motion only)

**VALUE\_ANGULAR** Evoked when angular order had been changed (used in steerable motion only)

**VALUE\_SIGMA** Evoked when Gaussian standard deviation had been changed (used in steerable motion only)

**VALUE\_SOURCE\_FRAME\_RATE** Evoked when source frame rate had been changed

**VALUE\_PROCESS\_FRAME\_RATE** Evoked when processing frame rate had been changed

**VALUE\_FILE\_DURATION** Evoked when get video file length

**VALUE\_FILE\_PROGRESS** Evoked during file processing, return the timestamp in video file

**VALUE\_CAMERA\_COUNT** Evoked when camera count had been changed

**AUTOFOCUS\_AVAILABLE** Evoked after source opened, indicate whether the source can do auto focus

**AUTOFOCUS\_ENABLED** Evoked after source opened, indicate whether auto focus is enabled

**OPEN\_CAMERA\_SUCCESS** Evoked when webcam opened successful

**OPEN\_CAMERA\_FAIL** Evoked when webcam opened fail

**OPEN\_FILE\_SUCCESS** Evoked when file opened successful

**OPEN\_FILE\_FAIL** Evoked when file opened fail

**OPEN\_EXTERNAL\_SOURCE\_SUCCESS** Evoked when external source opened successful

**OPEN\_EXTERNAL\_SOURCE\_FAIL** Evoked when external source opened fail

**CLOSE\_SOURCE\_SUCCESS** Evoked when source closed successful

**FILE\_ARCHIVED** Evoked when file archived

**END\_OF\_FILE** Evoked when reach to end of file

### 1.1.3.2 enum ProcessMethod

Process Algorithm

Enumerator

**LINEAR\_MOTION** Eulerian linear motion amplification method

**LINEAR\_COLOR** Eulerian linear color amplification method

**STEERABLE\_MOTION** Eulerian phase based motion amplification method, process in frequency domain

**RIESZ\_MOTION** Eulerian phase based motion amplification method, process in time domain

### 1.1.3.3 enum SourceOperation

Video Source

Enumerator

**CLOSE\_SOURCE** Close video source

**OPEN\_CAMERA\_SOURCE** Open webcam source

**OPEN\_FILE\_SOURCE** Open file source

**OPEN\_EXTERNAL\_SOURCE** Open user define source

#### 1.1.3.4 enum RoiMode

Region of Interest Mode Different way to extract ROI from a image

Enumerator

**MANUAL\_ROI** User can assign a ROI in the image

**FACE\_DETECT** Doing face detection first, then locate the fore head region as ROI

**FACE\_TRACK** Same as FACE\_DETECT, plus tracking technique

**MARKER\_DETECT** Doing marker detection, then locate the bottom of marker as ROI

The marker is rotation invariant, please refer to ArUco website

See Also

<http://www.uco.es/investigacion/grupos/ava/node/26>

#### 1.1.3.5 enum SourceAccess

Video source control/query type, only some of them are available

Enumerator

**CAMERA\_EXPOSURE** camera exposure access

**CAMERA\_AUTO\_FOCUS** camera auto focus access

**CAMERA\_FRAME\_RATE** camera frame rate access

**CAMERA\_BRIGHTNESS** camera brightness access

**CAMERA\_CONTRAST** camera contrast access

**CAMERA\_SATURATION** camera saturation access

**CAMERA\_AUTO\_WHITEBALANCE** camera whitebalance access

**CAMERA\_BACKLIGHT\_COMPENSATION** camera backlight compensation access

**CAMERA\_ABSOLUTE\_FOCUS** camera absolute focus access

**CAMERA\_ZOOM\_IN** camera zoom in

**CAMERA\_ZOOM\_OUT** camera zoom out

**CAMERA\_ZOOM\_RESET** camera zoom reset

**CAMERA\_PAN\_LEFT** camera pan left

**CAMERA\_PAN\_RIGHT** camera pan right

**CAMERA\_PAN\_RESET** camera pan reset

**CAMERA\_TILT\_UP** camera tilt up

**CAMERA\_TILT\_DOWN** camera tilt down

**CAMERA\_TILT\_RESET** camera tilt reset

**CAMERA\_PTZ\_RESET** reset pan, tilt and zoom

**FILE\_SEEK** Video file seek to some timestamp

**FILE\_PROGRESS** Video file current progress

**FILE\_DURATION** Video file length

**FILE\_PLAY** Play video file

**FILE\_PAUSE** Pause video file

**FILE\_FORWARD** Forward video file

**FILE\_BACKWARD** Backward video file

**FILE\_STOP** Stop video file

**FILE\_NORMAL\_PLAYBACK** Video file normal playback

**FILE\_TURBO\_PLAYBACK** Video file turbo playback

**IMAGE\_FLIP** Flip the input image

### 1.1.3.6 enum ParameterType

Algorithm parameters type

Enumerator

- ALPHA** the amplification factor
- HIGHCUT** high cutoff of temporal filter
- LOWCUT** low cutoff of temporal filter
- LEVEL** Gaussian pyramid level (used in linear color only)
- LEVEL\_BOUND** Maximum value for level
- LAMBDA** spatial frequency cutoff (used in linear motion only)
- LAMBDA\_BOUND** Maximum value for lambda
- RADIAL\_OCTAVE** number of octave in a spatial bands (used in steerable motion only)
- ANGULAR\_ORDER** number of orientation in the steerable pyramid (used in steerable motion only)
- SIGMA** the Gaussian standard deviation (used in steerable motion only)
- FRAMERATE** Processing frame rate
- CUTOFF\_BOUND** Maximum value for temporal filter cutoff

### 1.1.3.7 enum LicenseResponse

License verification response type

Enumerator

- LICENSE\_OFFLINE\_WARNING** license passed offline verification, but it has to finish an online verification in one month
- LICENSE\_OFFLINE\_SUCCESS** license passed offline verification
- LICENSE\_SUCCESS** license is valid
- LICENSE\_OFFLINE\_FORBIDDEN** license must take an online verification
- LICENSE\_MISSING** can not find any license
- LICENSE\_INCOMPLETE** incomplete license
- LICENSE\_EXPIRED** license is invalid

### 1.1.3.8 enum RecordType

Video record type

Enumerator

- RECORD\_NONE** Don't record video
- RECORD\_ORIGINAL\_ONLY** Only record original video
- RECORD\_PROCESSED\_ONLY** Only record processed video
- RECORD\_BOTH** Record both videos

## 1.1.4 Function Documentation

### 1.1.4.1 enum LicenseResponse initMagEngine ( CallbackFunction cb )

Initialize function.

be sure to call this function before doing anything

## Parameters

|           |   |
|-----------|---|
| <i>cb</i> | pass your CallBackFunction as a parameter |
|-----------|---|

## 1.1.4.2 void destroyMagEngine ( )

Release function.

be sure to call this function at the end of program to release the pointer

1.1.4.3 void setProcessMethod ( enum ProcessMethod *processMethod* )

Algorithm setting function.

set the processing algorithm you want to use

## Parameters

|                      |                       |
|----------------------|-----------------------|
| <i>processMethod</i> | the process algorithm |
|----------------------|-----------------------|

1.1.4.4 void setSource ( enum SourceOperation *operation*, const char \* *source* )

Video source setting.

set the video source from a file or webcam

if the device is a webcam, use videoX, X is the camera index

## Parameters

|                  |   |
|------------------|---|
| <i>operation</i> | specify the SourceOperation                       |
| <i>source</i>    | the name of the source (file name or device name) |

1.1.4.5 void setOutput ( const char \* *output* )

Output filename setting.

set the filename of output file

## Parameters

|               |                             |
|---------------|-----------------------------|
| <i>output</i> | specify the output filename |
|---------------|-----------------------------|

1.1.4.6 void setWebBackendMode ( bool *webBackendFlag* )

Toggle web backend mode.

use this function to toggle file output mode

## Parameters

|                        |  |
|------------------------|--|
| <i>webBackend-Flag</i> | specify whether to run as web backend mode |
|------------------------|--|

1.1.4.7 bool controlSource ( enum SourceAccess *type*, int *value* = 0 )

Camera or file control.

use this function to control camera or file source

## Parameters

|              |                                   |
|--------------|-----------------------------------|
| <i>type</i>  | specify control type SourceAccess |
| <i>value</i> | specify the value                 |

## Returns

whether the control is successful

## 1.1.4.8 int getCameraCount ( )

Get the number of connected cameras.

## Returns

the number of connected cameras

## 1.1.4.9 int64\_t querySource ( enum SourceAccess type )

Camera or file query.

query statics from video source

## Parameters

|             |                    |
|-------------|--------------------|
| <i>type</i> | specify query type |
|-------------|--------------------|

## Returns

the current value of the query type is return, if the type is unaccessible than UNAVAILABLE is return

## 1.1.4.10 void setROIMode ( enum RoiMode mode )

Region of interest method setting.

set to the specified ROI method

## Parameters

|             |                |
|-------------|----------------|
| <i>mode</i> | the ROI method |
|-------------|----------------|

## 1.1.4.11 enum RoiMode getROIMode ( )

Get the current ROI mode.

## Returns

the current chosen ROI mode

## 1.1.4.12 void setROIView ( uint16\_t x, uint16\_t y, uint16\_t width, uint16\_t height )

Set Region of interest.

only available in MANUAL\_ROI mode

## Parameters

|               |  |
|---------------|--|
| <i>x</i>      | the x coordinate of the top-left corner of ROI |
| <i>y</i>      | the y coordinate of the top-left corner of ROI |
| <i>width</i>  | the width of ROI                               |
| <i>height</i> | the height of ROI                              |

#### 1.1.4.13 void setFilterParameter ( enum ParameterType type, float value )

Filter parameter setting.

select a filter parameter to set to target value

##### Parameters

|              |                        |
|--------------|------------------------|
| <i>type</i>  | specify parameter type |
| <i>value</i> | target value           |

#### 1.1.4.14 void setFaceDectonXML ( std::string xmlFile )

Face detection XML selection.

Select which XML file you want to use for OpenCV face detection, they can be found in data/ folder

##### Parameters

|                |                               |
|----------------|-------------------------------|
| <i>xmlFile</i> | the file name of the XML file |
|----------------|-------------------------------|

#### 1.1.4.15 std::string getFaceDectonXML ( )

Get the name of the selected XML file.

##### Returns

the name of the current XML file

#### 1.1.4.16 void setFileOutput ( enum RecordType recordType )

Enable or disable file output.

set to RECORD\_NONE if you don't want to store any file

set to RECORD\_PROCESSED\_ONLY if you want to record processed result only

set to RECORD\_ORIGINAL\_ONLY if you want to record original source only

set to RECORD\_BOTH if you want to record both

##### Parameters

|                   |                         |
|-------------------|-------------------------|
| <i>recordType</i> | specify the record type |
|-------------------|-------------------------|

#### 1.1.4.17 enum ProcessMethod getProcessMethod ( )

Get the current process method.

retrun the algoeithm Geko is using now

##### Returns

the current chosen algorithm

#### 1.1.4.18 std::string getSourceName ( )

Get the video source name.

return file name when source is file

return videoX when source is webcam

#### Returns

current source name

#### 1.1.4.19 struct FilterParameter getFilterParameter ( )

Get the [FilterParameter](#) structure.

the [FilterParameter](#) structure contains the current parameter values

#### Returns

the current filter parameter struct

#### 1.1.4.20 struct GekoWriterState getFileOutputState ( )

Get the current info about writer.

#### Returns

the current writer state struct

#### 1.1.4.21 void setDataQueueLength ( size\_t len )

set the maximum data queue length for rate calculation

set the maximum data length you want to use for rate calculation

NOTE 1: You have to use this function or the `getDataQueueLength` function at least once to enable the rate calculation function and callback

NOTE 2: Since we use DFT to calculate rate, the performance is best if  $len = 2^a * 3^b * 5^c \dots$ ,

if your `len` didn't follow the rule, we will calculate an optimal size greater than or equal to `len` for you

#### Parameters

|            |  |
|------------|--|
| <i>len</i> | set the amount of data to calculate rate |
|------------|--|

#### 1.1.4.22 size\_t getDataQueueLength ( )

get the currently maximum data queue length

return the currently maximum data queue length

if you didn't use `setDataQueueLength` before, the function will return `NULLQUEUE`

otherwise it will return the optimized length  $2^a * 3^b * 5^c \dots$

#### Returns

the maximum data queue length

## 1.2 MediaSourceBase

### Classes

- class [MediaSourceBase](#)

### 1.2.1 Detailed Description

#### Author

Vince

#### Date

27 Feb 2014

The base class for media source  
please implement this class if you want to use your own source

## 2 Class Documentation

### 2.1 EventValue Struct Reference

```
#include <MagEngineAPI.h>
```

#### Public Types

- enum [ValueType](#) {  
**NonValue, IntegerValue, FloatValue, StringValue,**  
**FrameValue, FloatDataQueueue** }

#### Public Member Functions

- [EventValue](#) (enum [MagEngineEvent](#) evt=[NONE\\_EVENT](#))  
*Constructor for no value event type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, bool i)  
*Constructor for boolean type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, int i)  
*Constructor for integer type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, size\_t i)  
*Constructor for unsigned integer type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, int64\_t i)  
*Constructor for 64bits integer type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, float d)  
*Constructor for floating point type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, double d)  
*Constructor for double floating point type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, std::string s)  
*Constructor for string type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, char \*data, uint32\_t len, uint16\_t w, uint16\_t h)  
*Constructor for image buffer type.*
- [EventValue](#) (enum [MagEngineEvent](#) evt, float \*data, uint32\_t len)  
*Constructor for signal data queue.*

#### Public Attributes

- enum [MagEngineEvent](#) [event](#)
- enum [EventValue::ValueType](#) [valueType](#)
- std::string [sValue](#)
- union {  
double [dValue](#)  
int64\_t [iValue](#)  
};
- void \* [dataBuffer](#)
- uint16\_t [imageWidth](#)
- uint16\_t [imageHeight](#)

### 2.1.1 Detailed Description

This is the structure of callback event value

Callback event have different value types

You have to check the value type before any access

or you may get wrong value (e.g. the callback value is string but you accsee integer)

For implementation please refer to LambdaVueConsole.cpp

#### See Also

LambdaVueConsole.cpp

### 2.1.2 Member Enumeration Documentation

#### 2.1.2.1 enum `EventValue::ValueType`

The callback event value type, Always check value type before any accessing

### 2.1.3 Member Data Documentation

#### 2.1.3.1 enum `MagEngineEvent` `EventValue::event`

Event index, indicate the type of event

#### 2.1.3.2 `std::string` `EventValue::sValue`

String value type

#### 2.1.3.3 `double` `EventValue::dValue`

Double value type

#### 2.1.3.4 `int64_t` `EventValue::iValue`

`int64_t` value type

#### 2.1.3.5 `void*` `EventValue::dataBuffer`

pointer to data buffer, can be a image or a queue of signal data

#### 2.1.3.6 `uint16_t` `EventValue::imageWidth`

image width

#### 2.1.3.7 `uint16_t` `EventValue::imageHeight`

image height

## 2.2 FilterParameter Struct Reference

```
#include <MagEngineAPI.h>
```

#### Public Attributes

- `size_t` **alpha**

- float **highcut**
- float **lowcut**
- float **lambda**
- size\_t **level**
- size\_t **radialOctave**
- size\_t **angularOrder**
- size\_t **gaussianSigma**
- float **framerate**
- float **lambdaBound**
- float **cutOffBound**

### 2.2.1 Detailed Description

Algorithm parameters structure, store all the parameter value

## 2.3 GekoWriterState Struct Reference

```
#include <MagEngineAPI.h>
```

### Public Attributes

- enum [RecordType](#) **recordType**
- int [processedWidth](#)
- int [processedHeight](#)
- std::string [processedDestFileName](#)
- int [originalWidth](#)
- int [originalHeight](#)
- std::string [originalDestFileName](#)

### 2.3.1 Detailed Description

Current video writer status

### 2.3.2 Member Data Documentation

#### 2.3.2.1 int GekoWriterState::processedWidth

processed video width

#### 2.3.2.2 int GekoWriterState::processedHeight

processed video height

#### 2.3.2.3 std::string GekoWriterState::processedDestFileName

name of the processed video file

#### 2.3.2.4 int GekoWriterState::originalWidth

original video width

#### 2.3.2.5 int GekoWriterState::originalHeight

original video height

### 2.3.2.6 std::string GekoWriterState::originalDestFileName

name of the original video file

## 2.4 MediaSourceBase Class Reference

### Public Member Functions

- virtual void `destroy` ()=0  
*Destroy this media source.*
- virtual std::string `getSourceName` ()=0  
*Get the video source name.*
- virtual bool `switchSource` (const char \*sourceURL)=0  
*Switch to the specified source name.*
- virtual void `closeSource` ()=0  
*Close this media source.*
- virtual void `getImage` (char \*buf, uint32\_t &len, uint16\_t &width, uint16\_t &height, int64\_t &timestamp)=0  
*Assign your frame data so Geko can process it.*
- virtual size\_t `getFrameCount` ()=0  
*Return the amount of frame in this media source.*
- virtual size\_t `getWidth` ()=0  
*The image width of this media source.*
- virtual size\_t `getHeight` ()=0  
*The image height of this media source.*
- virtual float `getFrameRate` ()  
*Return the frame rate of this media source.*
- virtual int `getCameraCount` ()  
*Return the number of connected cameras.*
- virtual bool `isValid` ()=0  
*Return whether the source is valid.*
- virtual bool `isFile` ()  
*Return whether this media source is a file.*
- virtual int64\_t `query` (enum `geko::SourceAccess`)  
*Return the specify info of this media source.*
- virtual bool `control` (enum `geko::SourceAccess`, int)  
*Set the specify control type of media source.*

### 2.4.1 Member Function Documentation

#### 2.4.1.1 virtual void MediaSourceBase::destroy ( ) [pure virtual]

Destroy this media source.

implement how your media source should be destroyed

#### 2.4.1.2 virtual std::string MediaSourceBase::getSourceName ( ) [pure virtual]

Get the video source name.

#### Returns

the name of this media source

2.4.1.3 `virtual bool MediaSourceBase::switchSource ( const char * sourceURL ) [pure virtual]`

Switch to the specified source name.

implement the way to switch your media source from current video source to the specified video source

if the device is a webcam, its name should be videoX, where X is the camera index

Parameters

|                  |   |
|------------------|---|
| <i>sourceURL</i> | the name of the source (file name or device name) |
|------------------|---|

Returns

whether the switch is successful

2.4.1.4 `virtual void MediaSourceBase::closeSource ( ) [pure virtual]`

Close this media source.

implement how your media source should be closed

2.4.1.5 `virtual void MediaSourceBase::getImage ( char * buf, uint32_t & len, uint16_t & width, uint16_t & height, int64_t & timestamp ) [pure virtual]`

Assign your frame data so Geko can process it.

put your input image data in framebuf, and specify its width, height and timestamp

Parameters

|                  |   |
|------------------|---|
| <i>framebuf</i>  | put your image data in here, the image color space should be YUV420 |
| <i>width</i>     | the width of the image  |
| <i>height</i>    | the height of the image   |
| <i>timestamp</i> | image timestamp, used for file recording                            |

2.4.1.6 `virtual size_t MediaSourceBase::getFrameCount ( ) [pure virtual]`

Return the amount of frame in this media source.

Returns

the total number of frame in this media source

2.4.1.7 `virtual size_t MediaSourceBase::getWidth ( ) [pure virtual]`

The image width of this media source.

Returns

the image width

2.4.1.8 `virtual size_t MediaSourceBase::getHeight ( ) [pure virtual]`

The image height of this media source.

Returns

the image height

2.4.1.9 `virtual float MediaSourceBase::getFrameRate ( ) [virtual]`

Return the frame rate of this media source.

**Returns**

the frame rate

2.4.1.10 `virtual int MediaSourceBase::getCameraCount ( ) [virtual]`

Return the number of connected cameras.

**Returns**

the number of connected cameras

2.4.1.11 `virtual bool MediaSourceBase::isValid ( ) [pure virtual]`

Return whether the source is valid.

implement your method to check whether this media source is valid or not

**Returns**

whether this media source is valid

2.4.1.12 `virtual bool MediaSourceBase::isFile ( ) [virtual]`

Return whether this media source is a file.

**Returns**

true if this source is a file, false otherwise

2.4.1.13 `virtual int64_t MediaSourceBase::query ( enum geko::SourceAccess ) [virtual]`

Return the specify info of this media source.

please define how your source will return the source information define in SourceAccess

**Returns**

should return the value of the specified control type, please return UNAVAILABLE when the access is unavailable

2.4.1.14 `virtual bool MediaSourceBase::control ( enum geko::SourceAccess , int ) [virtual]`

Set the specify control type of media source.

please define how to set your media source

**Returns**

should return whether the control is successful

## Index

- ALPHA
  - MagEngineAPI, 7
- ANGULAR\_ORDER
  - MagEngineAPI, 7
- AUTOFOCUS\_AVAILABLE
  - MagEngineAPI, 5
- AUTOFOCUS\_ENABLED
  - MagEngineAPI, 5
  
- CAMERA\_ABSOLUTE\_FOCUS
  - MagEngineAPI, 6
- CAMERA\_AUTO\_FOCUS
  - MagEngineAPI, 6
- CAMERA\_AUTO\_WHITEBALANCE
  - MagEngineAPI, 6
- CAMERA\_BACKLIGHT\_COMPENSATION
  - MagEngineAPI, 6
- CAMERA\_BRIGHTNESS
  - MagEngineAPI, 6
- CAMERA\_CONTRAST
  - MagEngineAPI, 6
- CAMERA\_EXPOSURE
  - MagEngineAPI, 6
- CAMERA\_FRAME\_RATE
  - MagEngineAPI, 6
- CAMERA\_PAN\_LEFT
  - MagEngineAPI, 6
- CAMERA\_PAN\_RESET
  - MagEngineAPI, 6
- CAMERA\_PAN\_RIGHT
  - MagEngineAPI, 6
- CAMERA\_PTZ\_RESET
  - MagEngineAPI, 6
- CAMERA\_SATURATION
  - MagEngineAPI, 6
- CAMERA\_TILT\_DOWN
  - MagEngineAPI, 6
- CAMERA\_TILT\_RESET
  - MagEngineAPI, 6
- CAMERA\_TILT\_UP
  - MagEngineAPI, 6
- CAMERA\_ZOOM\_IN
  - MagEngineAPI, 6
- CAMERA\_ZOOM\_OUT
  - MagEngineAPI, 6
- CAMERA\_ZOOM\_RESET
  - MagEngineAPI, 6
- CLOSE\_SOURCE
  - MagEngineAPI, 5
- CLOSE\_SOURCE\_SUCCESS
  - MagEngineAPI, 5
- CUTOFF\_BOUND
  - MagEngineAPI, 7
- CallBackFunction
  - MagEngineAPI, 4
  
- closeSource
  - MediaSourceBase, 17
- control
  - MediaSourceBase, 18
- controlSource
  - MagEngineAPI, 8
  
- dValue
  - EventValue, 14
- dataBuffer
  - EventValue, 14
- destroy
  - MediaSourceBase, 16
- destroyMagEngine
  - MagEngineAPI, 8
  
- END\_OF\_FILE
  - MagEngineAPI, 5
- event
  - EventValue, 14
- EventValue, 13
  - dValue, 14
  - dataBuffer, 14
  - event, 14
  - iValue, 14
  - imageHeight, 14
  - imageWidth, 14
  - sValue, 14
  - ValueType, 14
  
- FACE\_DETECT
  - MagEngineAPI, 6
- FACE\_IMAGE
  - MagEngineAPI, 4
- FACE\_TRACK
  - MagEngineAPI, 6
- FILE\_ARCHIVED
  - MagEngineAPI, 5
- FILE\_BACKWARD
  - MagEngineAPI, 6
- FILE\_DURATION
  - MagEngineAPI, 6
- FILE\_FORWARD
  - MagEngineAPI, 6
- FILE\_NORNAL\_PLAYBACK
  - MagEngineAPI, 6
- FILE\_PAUSE
  - MagEngineAPI, 6
- FILE\_PLAY
  - MagEngineAPI, 6
- FILE\_PROGRESS
  - MagEngineAPI, 6
- FILE\_SEEK
  - MagEngineAPI, 6
- FILE\_STOP
  - MagEngineAPI, 6

- FILE\_TURBO\_PLAYBACK
  - MagEngineAPI, 6
- FILTER\_CHANGED
  - MagEngineAPI, 4
- FRAMERATE
  - MagEngineAPI, 7
- FREQUENCY\_NUMBER
  - MagEngineAPI, 4
- FilterParameter, 14
- GekoWriterState, 15
  - originalDestFileName, 15
  - originalHeight, 15
  - originalWidth, 15
  - processedDestFileName, 15
  - processedHeight, 15
  - processedWidth, 15
- getCameraCount
  - MagEngineAPI, 9
  - MediaSourceBase, 18
- getDataQueueLength
  - MagEngineAPI, 11
- getFaceDetectionXML
  - MagEngineAPI, 10
- getFileOutputState
  - MagEngineAPI, 11
- getFilterParameter
  - MagEngineAPI, 11
- getFrameCount
  - MediaSourceBase, 17
- getFrameRate
  - MediaSourceBase, 17
- getHeight
  - MediaSourceBase, 17
- getImage
  - MediaSourceBase, 17
- getProcessMethod
  - MagEngineAPI, 10
- getROIMode
  - MagEngineAPI, 9
- getSourceName
  - MagEngineAPI, 10
  - MediaSourceBase, 16
- getWidth
  - MediaSourceBase, 17
- HIGHCUT
  - MagEngineAPI, 7
- IMAGE\_FLIP
  - MagEngineAPI, 6
- iValue
  - EventValue, 14
- imageHeight
  - EventValue, 14
- imageWidth
  - EventValue, 14
- initMagEngine
  - MagEngineAPI, 7
- isFile
  - MediaSourceBase, 18
- isValid
  - MediaSourceBase, 18
- LAMBDA
  - MagEngineAPI, 7
- LAMBDA\_BOUND
  - MagEngineAPI, 7
- LEVEL
  - MagEngineAPI, 7
- LEVEL\_BOUND
  - MagEngineAPI, 7
- LICENSE\_EXPIRED
  - MagEngineAPI, 7
- LICENSE\_INCOMPLETE
  - MagEngineAPI, 7
- LICENSE\_MISSING
  - MagEngineAPI, 7
- LICENSE\_OFFLINE\_FORBIDDEN
  - MagEngineAPI, 7
- LICENSE\_OFFLINE\_SUCCESS
  - MagEngineAPI, 7
- LICENSE\_OFFLINE\_WARNING
  - MagEngineAPI, 7
- LICENSE\_SUCCESS
  - MagEngineAPI, 7
- LINEAR\_COLOR
  - MagEngineAPI, 5
- LINEAR\_MOTION
  - MagEngineAPI, 5
- LOWCUT
  - MagEngineAPI, 7
- LicenseResponse
  - MagEngineAPI, 7
- MANUAL\_ROI
  - MagEngineAPI, 6
- MARKER\_DETECT
  - MagEngineAPI, 6
- MagEngineAPI
  - ALPHA, 7
  - ANGULAR\_ORDER, 7
  - AUTOFOCUS\_AVAILABLE, 5
  - AUTOFOCUS\_ENABLED, 5
  - CAMERA\_ABSOLUTE\_FOCUS, 6
  - CAMERA\_AUTO\_FOCUS, 6
  - CAMERA\_AUTO\_WHITEBALANCE, 6
  - CAMERA\_BACKLIGHT\_COMPENSATION, 6
  - CAMERA\_BRIGHTNESS, 6
  - CAMERA\_CONTRAST, 6
  - CAMERA\_EXPOSURE, 6
  - CAMERA\_FRAME\_RATE, 6
  - CAMERA\_PAN\_LEFT, 6
  - CAMERA\_PAN\_RESET, 6
  - CAMERA\_PAN\_RIGHT, 6
  - CAMERA\_PTZ\_RESET, 6
  - CAMERA\_SATURATION, 6
  - CAMERA\_TILT\_DOWN, 6

CAMERA\_TILT\_RESET, 6  
 CAMERA\_TILT\_UP, 6  
 CAMERA\_ZOOM\_IN, 6  
 CAMERA\_ZOOM\_OUT, 6  
 CAMERA\_ZOOM\_RESET, 6  
 CLOSE\_SOURCE, 5  
 CLOSE\_SOURCE\_SUCCESS, 5  
 CUTOFF\_BOUND, 7  
 END\_OF\_FILE, 5  
 FACE\_DETECT, 6  
 FACE\_IMAGE, 4  
 FACE\_TRACK, 6  
 FILE\_ARCHIVED, 5  
 FILE\_BACKWARD, 6  
 FILE\_DURATION, 6  
 FILE\_FORWARD, 6  
 FILE\_NORNAL\_PLAYBACK, 6  
 FILE\_PAUSE, 6  
 FILE\_PLAY, 6  
 FILE\_PROGRESS, 6  
 FILE\_SEEK, 6  
 FILE\_STOP, 6  
 FILE\_TURBO\_PLAYBACK, 6  
 FILTER\_CHANGED, 4  
 FRAMERATE, 7  
 FREQUENCY\_NUMBER, 4  
 HIGHCUT, 7  
 IMAGE\_FLIP, 6  
 LAMBDA, 7  
 LAMBDA\_BOUND, 7  
 LEVEL, 7  
 LEVEL\_BOUND, 7  
 LICENSE\_EXPIRED, 7  
 LICENSE\_INCOMPLETE, 7  
 LICENSE\_MISSING, 7  
 LICENSE\_OFFLINE\_FORBIDDEN, 7  
 LICENSE\_OFFLINE\_SUCCESS, 7  
 LICENSE\_OFFLINE\_WARNING, 7  
 LICENSE\_SUCCESS, 7  
 LINEAR\_COLOR, 5  
 LINEAR\_MOTION, 5  
 LOWCUT, 7  
 MANUAL\_ROI, 6  
 MARKER\_DETECT, 6  
 NO\_FACE\_IMAGE, 4  
 NONE\_EVENT, 4  
 OPEN\_CAMERA\_FAIL, 5  
 OPEN\_CAMERA\_SOURCE, 5  
 OPEN\_CAMERA\_SUCCESS, 5  
 OPEN\_EXTERNAL\_SOURCE, 5  
 OPEN\_EXTERNAL\_SOURCE\_FAIL, 5  
 OPEN\_EXTERNAL\_SOURCE\_SUCCESS, 5  
 OPEN\_FILE\_FAIL, 5  
 OPEN\_FILE\_SOURCE, 5  
 OPEN\_FILE\_SUCCESS, 5  
 ORIGINAL\_IMAGE, 4  
 PROCESSED\_IMAGE, 4  
 RADIAL\_OCTAVE, 7  
 RECORD\_BOTH, 7  
 RECORD\_NONE, 7  
 RECORD\_ORIGINAL\_ONLY, 7  
 RECORD\_PROCESSED\_ONLY, 7  
 RIESZ\_MOTION, 5  
 ROI\_CHANGED, 4  
 ROI\_RESET, 4  
 SIGMA, 7  
 SIGNAL\_DATA, 4  
 STEERABLE\_MOTION, 5  
 VALUE\_ALPHA, 4  
 VALUE\_ANGULAR, 5  
 VALUE\_CAMERA\_COUNT, 5  
 VALUE\_FILE\_DURATION, 5  
 VALUE\_FILE\_PROGRESS, 5  
 VALUE\_HIGHCUT, 5  
 VALUE\_LAMBDA, 5  
 VALUE\_LEVEL, 5  
 VALUE\_LOWCUT, 5  
 VALUE\_PROCESS\_FRAME\_RATE, 5  
 VALUE\_RADIAL, 5  
 VALUE\_SIGMA, 5  
 VALUE\_SOURCE\_FRAME\_RATE, 5  
 MagEngineAPI, 1  
   CallbackFunction, 4  
   controlSource, 8  
   destroyMagEngine, 8  
   getCameraCount, 9  
   getDataQueueLength, 11  
   getFaceDectonXML, 10  
   getFileOutputState, 11  
   getFilterParameter, 11  
   getProcessMethod, 10  
   getROIMode, 9  
   getSourceName, 10  
   initMagEngine, 7  
   LicenseResponse, 7  
   MagEngineEvent, 4  
   ParameterType, 6  
   ProcessMethod, 5  
   querySource, 9  
   RecordType, 7  
   RoiMode, 5  
   setDataQueueLength, 11  
   setFaceDectonXML, 10  
   setFileOutput, 10  
   setFilterParameter, 10  
   setOutput, 8  
   setProcessMethod, 8  
   setROIMode, 9  
   setROIView, 9  
   setSource, 8  
   setWebBackendMode, 8  
   SourceAccess, 6  
   SourceOperation, 5  
 MagEngineEvent  
   MagEngineAPI, 4  
 MediaSourceBase, 12, 16

- closeSource, 17
- control, 18
- destroy, 16
- getCameraCount, 18
- getFrameCount, 17
- getFrameRate, 17
- getHeight, 17
- getImage, 17
- getSourceName, 16
- getWidth, 17
- isFile, 18
- isValid, 18
- query, 18
- switchSource, 16
- NO\_FACE\_IMAGE
  - MagEngineAPI, 4
- NONE\_EVENT
  - MagEngineAPI, 4
- OPEN\_CAMERA\_FAIL
  - MagEngineAPI, 5
- OPEN\_CAMERA\_SOURCE
  - MagEngineAPI, 5
- OPEN\_CAMERA\_SUCCESS
  - MagEngineAPI, 5
- OPEN\_EXTERNAL\_SOURCE
  - MagEngineAPI, 5
- OPEN\_EXTERNAL\_SOURCE\_FAIL
  - MagEngineAPI, 5
- OPEN\_EXTERNAL\_SOURCE\_SUCCESS
  - MagEngineAPI, 5
- OPEN\_FILE\_FAIL
  - MagEngineAPI, 5
- OPEN\_FILE\_SOURCE
  - MagEngineAPI, 5
- OPEN\_FILE\_SUCCESS
  - MagEngineAPI, 5
- ORIGINAL\_IMAGE
  - MagEngineAPI, 4
- originalDestFileName
  - GekoWriterState, 15
- originalHeight
  - GekoWriterState, 15
- originalWidth
  - GekoWriterState, 15
- PROCESSED\_IMAGE
  - MagEngineAPI, 4
- ParameterType
  - MagEngineAPI, 6
- ProcessMethod
  - MagEngineAPI, 5
- processedDestFileName
  - GekoWriterState, 15
- processedHeight
  - GekoWriterState, 15
- processedWidth
  - GekoWriterState, 15
- query
  - MediaSourceBase, 18
- querySource
  - MagEngineAPI, 9
- RADIAL\_OCTAVE
  - MagEngineAPI, 7
- RECORD\_BOTH
  - MagEngineAPI, 7
- RECORD\_NONE
  - MagEngineAPI, 7
- RECORD\_ORIGINAL\_ONLY
  - MagEngineAPI, 7
- RECORD\_PROCESSED\_ONLY
  - MagEngineAPI, 7
- RIESZ\_MOTION
  - MagEngineAPI, 5
- ROI\_CHANGED
  - MagEngineAPI, 4
- ROI\_RESET
  - MagEngineAPI, 4
- RecordType
  - MagEngineAPI, 7
- RoiMode
  - MagEngineAPI, 5
- SIGMA
  - MagEngineAPI, 7
- SIGNAL\_DATA
  - MagEngineAPI, 4
- STEERABLE\_MOTION
  - MagEngineAPI, 5
- sValue
  - EventValue, 14
- setDataQueueLength
  - MagEngineAPI, 11
- setFaceDectionXML
  - MagEngineAPI, 10
- setFileOutput
  - MagEngineAPI, 10
- setFilterParameter
  - MagEngineAPI, 10
- setOutput
  - MagEngineAPI, 8
- setProcessMethod
  - MagEngineAPI, 8
- setROIMode
  - MagEngineAPI, 9
- setROIView
  - MagEngineAPI, 9
- setSource
  - MagEngineAPI, 8
- setWebBackendMode
  - MagEngineAPI, 8
- SourceAccess
  - MagEngineAPI, 6
- SourceOperation
  - MagEngineAPI, 5
- switchSource

MediaSourceBase, [16](#)

VALUE\_ALPHA  
MagEngineAPI, [4](#)

VALUE\_ANGULAR  
MagEngineAPI, [5](#)

VALUE\_CAMERA\_COUNT  
MagEngineAPI, [5](#)

VALUE\_FILE\_DURATION  
MagEngineAPI, [5](#)

VALUE\_FILE\_PROGRESS  
MagEngineAPI, [5](#)

VALUE\_HIGHCUT  
MagEngineAPI, [5](#)

VALUE\_LAMBDA  
MagEngineAPI, [5](#)

VALUE\_LEVEL  
MagEngineAPI, [5](#)

VALUE\_LOWCUT  
MagEngineAPI, [5](#)

VALUE\_PROCESS\_FRAME\_RATE  
MagEngineAPI, [5](#)

VALUE\_RADIAL  
MagEngineAPI, [5](#)

VALUE\_SIGMA  
MagEngineAPI, [5](#)

VALUE\_SOURCE\_FRAME\_RATE  
MagEngineAPI, [5](#)

ValueType  
EventValue, [14](#)