Mvx2BasicIO

Generated by Doxygen 1.8.16

Chapter 1

Mantis Vision: Mvx2BasicIO

An extension module of *Mvx2* for file and network data accessing and storing.

Description

Mvx2BasicIO is a collection of classes and functions which together form an extension of the core *Mvx2* (*Mvx2* is documented in a dedicated document). The extension's emphasis is on working with Mvx2-formatted files and network streaming.

Following is a quick overview of the Mvx2BasicIO's purpose and features:

- provides graph nodes for accessing (reading and writing) Mvx2-formatted files,
- provides graph nodes for accessing (transmission and reception) Mvx2 network streams,
- provides utility for fast extraction of basic data information about Mvx2 files.

File Access

The extension provides multiple graph node implementations related to access to Mvx2-formatted files. The most basic are:

- Mvx2BasicIO::Mvx2FileReaderGraphNode for reading frame data from Mvx2-formatted files and
- Mvx2BasicIO::Mvx2FileWriterGraphNode for writing processed frames to Mvx2-formatted files.

Furthermore, the extension provides an utility class Mvx2BasicIO::Mvx2FileBasicDataInfo, which provides a fast access to the most basic queries related to content of Mvx2-formatted files (e.g. number of frames in a file), but it is also able to read the very first frame from the file, so any more advanced queries can be performed as well.

Networking

Another collection of classes of the extension adds support for streaming Mvx2 data over network:

- Mvx2BasicIO::NetworkReceiverGraphNode for reception of frame data from a network stream and
- Mvx2BasicIO::NetworkTransmitterGraphNode for transmission of processed frames via a network stream.

Details

Compiled using SuperNetwork plugin version 4.2.0 and MVX2File plugin version 3.3.1.

Chapter 2

Release Notes

1.0.0

Initial version.

Module

- **1.0.0_M1** | renamed NetworkTransmitterGraphNode::GetDroppedFramesCount() and NetworkTransmitterGraphNode::ResetDroppedFramesCounter() functions to MVGraphAPI::NetworkTransmitterGraphNode::GetDroppedAtomsCount() "NetworkTransmitterGraphNode::GetDroppedAtomsCount()" and MVGraphAPI::NetworkTransmitterGraphNode::ResetDroppedAtomsCounter() "NetworkTransmitterGraphNode::ResetDroppedAtomsCounter()" respectively

Documentation

- **1.0.0_D1** | added 'release notes' section
- **1.0.0_D2** | added/updated missing API reference documentation
- **1.0.0_D3** | switched documentation from xml-style comments to doxygen-style comments

Build support

- **1.0.0_BS1** | introduced MVGraph_SimpleAPIConfig.cmake, MVGraph_SimpleAPINetConfig.cmake and MVGraph_SimpleAPINet_iOSConfig.cmake

Samples

- **1.0.0_S1** | introduced MVGraph_SimpleAPIDemo and MVGraph_SimpleAPINetDemo samples for showcasing usage of MVGraph_SimpleAPI extension of MVGraphAPI (both samples are compiled using cmake and include python scripts for their simple compilation and execution)

2.0.0

Module

- **2.0.0_M1** | updated `Mvx2` 3rdparty dependency to version 3.0.0
- **2.0.0_M2** | updated `SuperNetwork` plugin to 2.0.0
- **2.0.0_M3** | updated `MVX2File` plugin to 2.0.0
- **2.0.0_M4** | introduced MVGraphAPI::Mvx2FileAsyncWriterGraphNode "Mvx2FileAsyncWriterGraphNode", which performs writing operation from standalone writing thread asynchronously, as an alternative to M↩ VGraphAPI::Mvx2FileWriterGraphNode "Mvx2FileWriterGraphNode"

Build support

- **2.0.0_BS1** | size of Android and LuminOS libraries reduced by ∼90%
- **2.0.0_BS2** | android API level raised from 19 to 21
- **2.0.0_BS3** | Linux and MacOS binaries do not consist of a versioned library file and a version-neutral symlink file anymore - the library file itself has version-neutral name

3.0.0

Module

- **3.0.0_M1** | updated `Mvx2` 3rdparty dependency to version 4.0.0
- **3.0.0_M2** | updated `SuperNetwork` plugin to 3.0.0
- **3.0.0_M3** | updated `MVX2File` plugin to 3.0.0
- **3.0.0_M4** | removed MVGraphAPI::AutoCompressorGraphNode and MVGraphAPI::AutoDecompressor↩ GraphNode since `Mvx2` now contains their alternatives
- **3.0.0_M5** | removed MVGraphAPI::InjectFileDataGraphNode and MVGraphAPI::InjectMemoryDataGraph↩ Node since `Mvx2` now contains their alternatives
- **3.0.0_M6** | removed MVGraphAPI::MeshData and MVGraphAPI::MeshSplitter since `Mvx2` now contains their alternatives
- **3.0.0_M7** | removed MVGraphAPI::SimpleDataLayersGuids since `Mvx2` now contains its alternative
- **3.0.0_M8** | removed frame data extractors since `Mvx2` now contains their alternatives:
  - MVGraphAPI::FrameAudioExtractor
  - MVGraphAPI::FrameMeshExtractor
  - MVGraphAPI::FrameMiscDataExtractor
  - MVGraphAPI::FrameTextureExtractor
- **3.0.0_M9** | renamed MVGraph_SimpleAPI module to Mvx2BasicIO:
  1. `MVGraph_SimpleAPI` product renamed to `Mvx2BasicIO`
  2. public header files of MVGraph_SimpleAPI moved to `include/Mvx2BasicIO` directory
  3. `MVGraphAPI` namespace renamed to `Mvx2BasicIO`
  4. `MVGraph_SimpleAPI.zip` file containing MVGraph_SimpleAPI/Mvx2BasicIO documentation renamed to `Mvx2BasicIO.zip`

5. updated Mvx2BasicIO documentation
6. introduced Mvx2BasicIO's own export macro `MVX2BASICIO_API` defined in file `Mvx2BasicIO/Mvx2BasicIO.h` instead of reusing Mvx2's `MVX2_API`
7. `MVGraph_SimpleAPIConfig.cmake` cmake-build file updated and renamed to `Mvx2BasicIOConfig.cmake`

- **3.0.0_M10** | renamed MVGraph_SimpleAPINet module to Mvx2BasicIONet:

   1. `MVGraph_SimpleAPI` product renamed to `Mvx2BasicIO`
   2. `MVGraphAPI` namespace renamed to `Mvx2BasicIO`
   3. `MVGraph_SimpleAPINet.zip` file containing MVGraph_SimpleAPINet/Mvx2BasicIONet documentation renamed to `Mvx2BasicIONet.zip`
   4. updated Mvx2BasicIONet documentation
   5. `MVGraphAPI::MVGraph_SimpleAPINetConstants` class renamed to `Mvx2BasicIO::Constants` and its `MV_GRAPH_SIMPLE_API_INTEROP_DLL` field to `INTEROP_DLL`
   6. `MVGraph_SimpleAPINetConfig.cmake` and `MVGraph_SimpleAPINet_iOSConfig.cmake` cmake-build files updated and renamed to `Mvx2BasicIONetConfig.cmake` and `Mvx2BasicIONet_iOSConfig.cmake` respectively

- **3.0.0_M11** | renamed `Mvx2BasicIO::Mvx2FileSimpleDataInfo` class to `Mvx2BasicIO::Mvx2FileBasicDat`

Samples

- **3.0.0_S1** | renamed `MVGraph_SimpleAPIDemo` and `MVGraph_SimpleAPINetDemo` to `Mvx2BasicIODemo` and `Mvx2BasicIONetDemo` respectively
- **3.0.0_S2** | updated sources and support scripts of `Mvx2BasicIODemo` and `Mvx2BasicIONetDemo` for latest Mvx2(Net) and Mvx2BasicIO(Net)

3.1.0

Module

- **3.1.0_M1** | updated `SuperNetwork` plugin to 4.0.0 (experimental IPv6 support)
- **3.1.0_M2** | added an option to enable experimental IPv6 support in Mvx2BasicIO::NetworkTransmitterGraphNode and Mvx2BasicIO::NetworkReceiverGraphNode graph nodes:

   - the feature may not work on all platforms as expected and may even prevent correct functioning of IPv4 communication
   - introduced `enableIPv6` parameter to Mvx2BasicIO::NetworkTransmitterGraphNode constructors with `false` as default value
   - introduced `enableIPv6` parameter to Mvx2BasicIO::NetworkTransmitterGraphNode::SetSockets with `false` as default value
   - introduced `enableIPv6` parameter to Mvx2BasicIO::NetworkReceiverGraphNode constructors with `false` as default value
   - introduced `enableIPv6` parameter to Mvx2BasicIO::NetworkReceiverGraphNode::SetSockets with `false` as default value

4.0.0

Module

- **4.0.0_M1** | updated `MVCommon` 3rdparty dependency to version 3.0.0
- **4.0.0_M2** | updated `Mvx2` 3rdparty dependency to version 5.0.0
- **4.0.0_M3** | updated `SuperNetwork` plugin to 4.1.0
- **4.0.0_M4** | updated `MVX2File` plugin to 3.1.0

Build support

- **4.0.0_BS1** | CMake minimal required version increased from 3.9 to 3.14
  - updated `Mvx2BasicIOConfig.cmake`, `Mvx2BasicIONetConfig.cmake` and `Mvx2BasicIONet_iOSConfig.cmake` scripts and their dependencies

Samples

- **4.0.0_S1** | extended `Mvx2BasicIODemo` and `Mvx2BasicIONetDemo` samples to print data profiles of frames during the inspection (a new feature introduced to `Mvx2` with version 5.0.0)
- **4.0.0_S2** | CMake minimal required version increased from 3.9 to 3.14
  - updated `CMakeLists.txt` of `Mvx2BasicIODemo` sample
  - updated `CMakeLists.txt` of `Mvx2BasicIONetDemo` sample
- **4.0.0_S3** | updated `Mvx2BasicIODemo` and `Mvx2BasicIONetDemo` samples' `CMakeLists.txt` and `make.py` scripts to expect MVCommon and Mvx2 dependencies and MVX2File and SuperNetwork plugins on a potentially different path than Mvx2BasicIO dependency
  - introduced `build/local_config/mvcommon_root_dir.cfg` config files inside the samples root directories, which shall specify a path to the MVCommon root directory
  - introduced `build/local_config/mvx2_root_dir.cfg` config files inside the samples root directories, which shall specify a path to the Mvx2 root directory
  - introduced `build/local_config/mvx2file_root_dir.cfg` config files inside the samples root directories, which shall specify a path to the MVX2File root directory
  - introduced `build/local_config/supernetwork_root_dir.cfg` config files inside the samples root directories, which shall specify a path to the SuperNetwork root directory

5.0.0

Module

- **5.0.0_M1** | updated `MVCommon` 3rdparty dependency to version 4.0.0
- **5.0.0_M2** | updated `Mvx2` 3rdparty dependency to version 6.0.0
- **5.0.0_M3** | updated `SuperNetwork` plugin to 4.2.0
- **5.0.0_M4** | updated `MVX2File` plugin to 3.3.0

Build support

- **5.0.0_BS1** | from now on the windows libraries are compiled using msvc compiler version 142 (VS 2019)
- **5.0.0_BS2** | upgraded `cmake/toolchains/ios.cmake` toolchain file used for building for iOS platform

Documentation

- **5.0.0_D1** | introduced PDF documentation as an alternative to the HTML one:
  - `doc/Mvx2BasicIO.pdf`
  - `doc/Mvx2BasicIONet.pdf`

Samples

- **5.0.0_S1** | from now on the windows libraries of the samples are compiled using msvc compiler version 142 (VS 2019)

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

Data Structure Documentation

5.1 Mvx2BasicIO::Mvx2FileAsyncReader Class Reference

A sequential reader of MVX2 files with an asynchronous access to processed frames.

`#include <Mvx2FileAsyncReader.h>`

Inherits NonAssignable.

Public Member Functions

- MVX2BASICIO_API Mvx2FileAsyncReader (MVCommon::String const &filePath, Mvx2API::FrameListener ∗pFrameListener, float fps=-1.0f)

  A constructor.
- MVX2BASICIO_API ~Mvx2FileAsyncReader ()

  A destructor.
- MVX2BASICIO_API bool Play (Mvx2API::RunnerPlaybackMode playbackMode, bool blockUntil↩ Stopped=false)

  Starts the file reading.
- MVX2BASICIO_API bool Stop ()

  Stops the file reading.

Static Public Attributes

- static const float FPS_MAX = 0.0f

  A special framerate value indicating that the maximal possible framerate shall be used.
- static const float FPS_FROM_SOURCE = -1.0f

  A special framerate value indicating that the framerate of an open source shall be used.
- static const float FPS_HALF_FROM_SOURCE = -2.0f

  A special framerate value indicating that the half of the framerate of an open source shall be used.
- static const float FPS_DOUBLE_FROM_SOURCE = -3.0f

  A special framerate value indicating that the double of the framerate of an open source shall be used.

5.1.1 Detailed Description

A sequential reader of MVX2 files with an asynchronous access to processed frames.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Mvx2FileAsyncReader()

```
Mvx2BasicIO::Mvx2FileAsyncReader::Mvx2FileAsyncReader (
            MVCommon::String const & filePath,
            Mvx2API::FrameListener * pFrameListener,
            float fps = -1.0f )  [inline]
```

A constructor.

Parameters

filePath	a path of the MVX2 file to read
pFrameListener	an asynchronous frames listener
fps	a framerate to follow

Exceptions

std::runtime_error	raised in case the internal graph could not be created

5.1.3 Member Function Documentation

5.1.3.1 Play()

```
bool Mvx2BasicIO::Mvx2FileAsyncReader::Play (
            Mvx2API::RunnerPlaybackMode playbackMode,
            bool blockUntilStopped = false )  [inline]
```

Starts the file reading.

Can be executed synchronously in case `blockUntilStopped` is set to true, or asynchronously when set to false.

Parameters

playbackMode	a playback mode of the reading
blockUntilStopped	an indication whether to block the call until the reading is stopped implicitly

Returns

true if the file reading successfully started

5.1.3.2 Stop()

```
bool Mvx2BasicIO::Mvx2FileAsyncReader::Stop ( )  [inline]
```

Stops the file reading.

Returns

true if the file reading successfully stopped

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/util/Mvx2FileAsyncReader.h

5.2 Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode Class Reference

A target graph node for asynchronous writing frames to an MVX2 file.

`#include <Mvx2FileAsyncWriterGraphNode.h>`

Inherits SingleFilterGraphNode.

Public Types

- enum FullBehaviour { FB_DROP_FRAMES, FB_BLOCK_FRAMES }

  Enumeration of supported behaviours when the buffer of frames is full.

Public Member Functions

- MVX2BASICIO_API Mvx2FileAsyncWriterGraphNode (MVCommon::String const &filePath, bool enableRecording=true, uint32_t bufferSize=3, FullBehaviour fullBehaviour=FB_DROP_FRAMES)

  A constructor.
- virtual MVX2BASICIO_API ∼Mvx2FileAsyncWriterGraphNode ()

  A destructor.
- MVX2BASICIO_API void EnableRecording (bool enable=true) const

  Enables/disables actual recording to the MVX2 file.
- MVX2BASICIO_API void SetFilePath (MVCommon::String const &filePath) const

  Changes the path of the MVX2 file to write to.
- MVX2BASICIO_API void SetFullBehaviour (FullBehaviour fullBehaviour)

  Sets a full-behaviour - action to perform when the buffer of frames becomes full.
- MVX2BASICIO_API uint64_t GetDroppedFramesCount () const

  Gets a value of internal counter of dropped frames.
- MVX2BASICIO_API void ResetDroppedFramesCounter () const

  Resets the internal counter of dropped frames to zero.

5.2.1 Detailed Description

A target graph node for asynchronous writing frames to an MVX2 file.

Asynchronous writing means that frames are pushed to a buffer from the pipeline thread and are pulled from the buffer and written to a file from a standalone writing thread.

Internally maintains a single writing filter. The same filter is reused even when the graph node is added to multiple graphs.

5.2.2 Member Enumeration Documentation

5.2.2.1 FullBehaviour

```
enum Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode::FullBehaviour
```

Enumeration of supported behaviours when the buffer of frames is full.

Enumerator

FB_DROP_FRAMES	When buffer is full, new frames from pipeline are dropped and pipeline continues its execution.
FB_BLOCK_FRAMES	When buffer is full, pipeline thread is blocked until there is some free space in the buffer.

5.2.3 Constructor & Destructor Documentation

5.2.3.1 Mvx2FileAsyncWriterGraphNode()

```
MVX2BASICIO_API Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode::Mvx2FileAsyncWriterGraphNode (
             MVCommon::String const & filePath,
             bool enableRecording = true,
             uint32_t bufferSize = 3,
             FullBehaviour fullBehaviour = FB_DROP_FRAMES )
```

A constructor.

Parameters

filePath	a path of the MVX2 file to write to
enableRecording	an indication whether the recording shall be enabled right away
bufferSize	a size of frames buffer
fullBehaviour	an initial full-behaviour

5.2.4 Member Function Documentation

5.2.4.1 EnableRecording()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode::EnableRecording (
            bool enable = true ) const
```

Enables/disables actual recording to the MVX2 file.

Parameters

enable	true in order to enable recording, false in order to disable it

5.2.4.2 GetDroppedFramesCount()

```
MVX2BASICIO_API uint64_t Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode::GetDroppedFramesCount ( )
const
```

Gets a value of internal counter of dropped frames.

Returns

dropped frames count

5.2.4.3 SetFilePath()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode::SetFilePath (
            MVCommon::String const & filePath ) const
```

Changes the path of the MVX2 file to write to.

Parameters

filePath	a new path of the MVX2 file

5.2.4.4 SetFullBehaviour()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileAsyncWriterGraphNode::SetFullBehaviour (
            FullBehaviour fullBehaviour )
```

Sets a full-behaviour - action to perform when the buffer of frames becomes full.

Parameters

fullBehaviour	a behaviour to set

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/graphnodes/Mvx2FileAsyncWriterGraphNode.h

5.3 Mvx2BasicIO::Mvx2FileBasicDataInfo Class Reference

A provider of basic data information about an MVX2 file.

```
#include <Mvx2FileBasicDataInfo.h>
```

Inherits NonAssignable.

Public Member Functions

- MVX2BASICIO_API Mvx2FileBasicDataInfo (MVCommon::String const &filePath)

  A constructor.
- virtual MVX2BASICIO_API ∼Mvx2FileBasicDataInfo ()

  A destructor.
- MVX2BASICIO_API bool IsValid () const

  Indicates whether the specified file is a valid MVX2 file.
- MVX2BASICIO_API bool IsSingleFrame () const

  Indicates whether the MVX2 file contains only a single frame.
- MVX2BASICIO_API uint32_t GetNumFrames () const

  Returns a number of frames in the MVX2 file.
- MVX2BASICIO_API float GetFPS () const

  Returns framerate of the MVX2 file.
- MVX2BASICIO_API bool HasDepthMap () const

  Checks the presence of depth map data in the MVX2 file.
- MVX2BASICIO_API bool HasIRTexture () const

  Checks the presence of IR texture data in the MVX2 file.
- MVX2BASICIO_API bool HasColorTexture () const

  Checks the presence of a color texture data in the MVX2 file.
- MVX2BASICIO_API bool HasVertices () const

  Checks the presence of vertex positions data in the MVX2 file.
- MVX2BASICIO_API bool HasNormals () const

  Checks the presence of vertex normals data in the MVX2 file.
- MVX2BASICIO_API bool HasColors () const

  Checks the presence of vertex colors data in the MVX2 file.
- MVX2BASICIO_API bool HasUVs () const

  Checks the presence of vertex UVs data in the MVX2 file.
- MVX2BASICIO_API bool HasIndices () const

  Checks the presence of vertex indices data in the MVX2 file.

- MVX2BASICIO_API bool HasAudio () const

  Checks the presence of audio data in the MVX2 file.
- MVX2BASICIO_API Mvx2API::Frame ∗ GetFirstFrame () const

  Returns the first frame of the MVX2 file.
- MVX2BASICIO_API bool CanRenderThumbnail () const

  Indicates whether it is possible to render a thumbnail image of the MVX2 file.
- MVX2BASICIO_API void RenderThumbnail (uint8_t ∗targetBufferRGBA, uint32_t targetWidth, uint32_t targetHeight) const

  Renders a thumbnail image of the MVX2 file.
- MVX2BASICIO_API uint8_t ∗ CreateAndRenderThumbnail (uint32_t targetWidth, uint32_t targetHeight) const

  Allocates a new image buffer and renders a thumbnail image of the MVX2 file to it.

5.3.1 Detailed Description

A provider of basic data information about an MVX2 file.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Mvx2FileBasicDataInfo()

```
MVX2BASICIO_API Mvx2BasicIO::Mvx2FileBasicDataInfo::Mvx2FileBasicDataInfo (
             MVCommon::String const & filePath )
```

A constructor.

Parameters

filePath	a path to the MVX2 file to provide info about

5.3.3 Member Function Documentation

5.3.3.1 CanRenderThumbnail()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::CanRenderThumbnail ( ) const
```

Indicates whether it is possible to render a thumbnail image of the MVX2 file.

Returns

true if the thumbnail image is available

5.3.3.2 CreateAndRenderThumbnail()

```
MVX2BASICIO_API uint8_t* Mvx2BasicIO::Mvx2FileBasicDataInfo::CreateAndRenderThumbnail (
            uint32_t targetWidth,
            uint32_t targetHeight ) const
```

Allocates a new image buffer and renders a thumbnail image of the MVX2 file to it.

Parameters

targetWidth	width of the image buffer
targetHeight	height of the image buffer

Returns

the new image buffer

5.3.3.3 GetFirstFrame()

```
MVX2BASICIO_API Mvx2API::Frame* Mvx2BasicIO::Mvx2FileBasicDataInfo::GetFirstFrame ( ) const
```

Returns the first frame of the MVX2 file.

Returns

the first frame

5.3.3.4 GetFPS()

```
MVX2BASICIO_API float Mvx2BasicIO::Mvx2FileBasicDataInfo::GetFPS ( ) const
```

Returns framerate of the MVX2 file.

Returns

framerate

5.3.3.5 GetNumFrames()

```
MVX2BASICIO_API uint32_t Mvx2BasicIO::Mvx2FileBasicDataInfo::GetNumFrames ( ) const
```

Returns a number of frames in the MVX2 file.

Returns

frames count

5.3.3.6 HasAudio()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasAudio ( ) const
```

Checks the presence of audio data in the MVX2 file.

Returns

true if the audio data are available

5.3.3.7 HasColors()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasColors ( ) const
```

Checks the presence of vertex colors data in the MVX2 file.

Returns

true if the vertex colors data are available

5.3.3.8 HasColorTexture()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasColorTexture ( ) const
```

Checks the presence of a color texture data in the MVX2 file.

Returns

true if a color texture data are available

5.3.3.9 HasDepthMap()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasDepthMap ( ) const
```

Checks the presence of depth map data in the MVX2 file.

Returns

true if the depth map data are available

5.3.3.10 HasIndices()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasIndices ( ) const
```

Checks the presence of vertex indices data in the MVX2 file.

Returns

true if the vertex indices data are available

5.3.3.11 HasIRTexture()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasIRTexture ( ) const
```

Checks the presence of IR texture data in the MVX2 file.

Returns

true if the IR texture data are available

5.3.3.12 HasNormals()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasNormals ( ) const
```

Checks the presence of vertex normals data in the MVX2 file.

Returns

true if the vertex normals data are available

5.3.3.13 HasUVs()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasUVs ( ) const
```

Checks the presence of vertex UVs data in the MVX2 file.

Returns

true if the vertex UVs data are available

5.3.3.14 HasVertices()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::HasVertices ( ) const
```

Checks the presence of vertex positions data in the MVX2 file.

Returns

true if the vertex positions data are available

5.3.3.15 IsSingleFrame()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::IsSingleFrame ( ) const
```

Indicates whether the MVX2 file contains only a single frame.

Returns

true if the file contains only a single frame

5.3.3.16 IsValid()

```
MVX2BASICIO_API bool Mvx2BasicIO::Mvx2FileBasicDataInfo::IsValid ( ) const
```

Indicates whether the specified file is a valid MVX2 file.

Returns

true if the file is a valid MVX2 file

5.3.3.17 RenderThumbnail()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileBasicDataInfo::RenderThumbnail (
             uint8_t * targetBufferRGBA,
             uint32_t targetWidth,
             uint32_t targetHeight ) const
```

Renders a thumbnail image of the MVX2 file.

Parameters

targetBufferRGBA	a pre-allocated buffer for the thumbnail image
targetWidth	width of the image buffer
targetHeight	height of the image buffer

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/util/Mvx2FileBasicDataInfo.h

5.4 Mvx2BasicIO::Mvx2FileRandomAccessReader Class Reference

A sequential reader of MVX2 files with a random access to processed frames.

```
#include <Mvx2FileRandomAccessReader.h>
```

Inherits NonAssignable.

Public Member Functions

- MVX2BASICIO_API Mvx2FileRandomAccessReader (MVCommon::String const &filePath)

  A constructor.
- MVX2BASICIO_API ∼Mvx2FileRandomAccessReader ()

  A destructor.
- MVX2BASICIO_API Mvx2API::Frame ∗ ReadFrame (uint32_t frameID)

  Reads a frame from the file.

5.4.1 Detailed Description

A sequential reader of MVX2 files with a random access to processed frames.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 Mvx2FileRandomAccessReader()

```
Mvx2BasicIO::Mvx2FileRandomAccessReader::Mvx2FileRandomAccessReader (
            MVCommon::String const & filePath )  [inline]
```

A constructor.

Parameters

filePath	a path of the MVX2 file to read

Exceptions

std::runtime_error	raised in case the internal graph could not be created

5.4.3 Member Function Documentation

5.4.3.1 ReadFrame()

```
Mvx2API::Frame * Mvx2BasicIO::Mvx2FileRandomAccessReader::ReadFrame (
            uint32_t frameID )  [inline]
```

Reads a frame from the file.

Parameters

frameID	an ID of the frame to read

Returns

a frame with the ID or nullptr if there is none

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/util/Mvx2FileRandomAccessReader.h

5.5 Mvx2BasicIO::Mvx2FileReaderGraphNode Class Reference

A source graph node for reading frames from an MVX2 file.

```
#include <Mvx2FileReaderGraphNode.h>
```

Inherits SingleFilterGraphNode.

Public Member Functions

- MVX2BASICIO_API Mvx2FileReaderGraphNode (MVCommon::String const &filePath)

  A constructor.
- virtual MVX2BASICIO_API ∼Mvx2FileReaderGraphNode ()

  A destructor.
- MVX2BASICIO_API void SetFilePath (MVCommon::String const &filePath) const

  Changes the path of the MVX2 file to read from.

5.5.1 Detailed Description

A source graph node for reading frames from an MVX2 file.

Internally maintains a single reading filter. The same filter is reused even when the graph node is added to multiple graphs.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Mvx2FileReaderGraphNode()

```
MVX2BASICIO_API Mvx2BasicIO::Mvx2FileReaderGraphNode::Mvx2FileReaderGraphNode (
            MVCommon::String const & filePath )
```

A constructor.

Parameters

filePath	a path of the MVX2 file to read from

5.5.3 Member Function Documentation

5.5.3.1 SetFilePath()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileReaderGraphNode::SetFilePath (
            MVCommon::String const & filePath ) const
```

Changes the path of the MVX2 file to read from.

Parameters

filePath	a new path of the MVX2 file

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/graphnodes/Mvx2FileReaderGraphNode.h

5.6 Mvx2BasicIO::Mvx2FileSyncReader Class Reference

A sequential reader of MVX2 files with a synchronous access to processed frames.

```
#include <Mvx2FileSyncReader.h>
```

Inherits NonAssignable.

Public Member Functions

- MVX2BASICIO_API Mvx2FileSyncReader (MVCommon::String const &filePath, Mvx2API::Runner↩ PlaybackMode playbackMode)

  A constructor.
- MVX2BASICIO_API ∼Mvx2FileSyncReader ()

  A destructor.
- MVX2BASICIO_API Mvx2API::Frame ∗ ReadNextFrame ()

  Reads next frame from the file.

5.6.1 Detailed Description

A sequential reader of MVX2 files with a synchronous access to processed frames.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 Mvx2FileSyncReader()

```
Mvx2BasicIO::Mvx2FileSyncReader::Mvx2FileSyncReader (
            MVCommon::String const & filePath,
            Mvx2API::RunnerPlaybackMode playbackMode )  [inline]
```

A constructor.

Parameters

filePath	a path of the MVX2 file to read
playbackMode	a playback mode of the reading

Exceptions

std::runtime_error	raised in case the internal graph could not be created

5.6.3 Member Function Documentation

5.6.3.1 ReadNextFrame()

```
Mvx2API::Frame * Mvx2BasicIO::Mvx2FileSyncReader::ReadNextFrame ( )  [inline]
```

Reads next frame from the file.

Returns

next processed frame or nullptr if there is none

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/util/Mvx2FileSyncReader.h

5.7 Mvx2BasicIO::Mvx2FileWriterGraphNode Class Reference

A target graph node for writing frames to an MVX2 file.

```
#include <Mvx2FileWriterGraphNode.h>
```

Inherits SingleFilterGraphNode.

Public Member Functions

- MVX2BASICIO_API Mvx2FileWriterGraphNode (MVCommon::String const &filePath, bool enable↩Recording=true)

  A constructor.
- virtual MVX2BASICIO_API ∼Mvx2FileWriterGraphNode ()

  A destructor.
- MVX2BASICIO_API void EnableRecording (bool enable=true) const

  Enables/disables actual recording to the MVX2 file.
- MVX2BASICIO_API void SetFilePath (MVCommon::String const &filePath) const

  Changes the path of the MVX2 file to write to.

5.7.1 Detailed Description

A target graph node for writing frames to an MVX2 file.

Internally maintains a single writing filter. The same filter is reused even when the graph node is added to multiple graphs.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Mvx2FileWriterGraphNode()

```
MVX2BASICIO_API Mvx2BasicIO::Mvx2FileWriterGraphNode::Mvx2FileWriterGraphNode (
            MVCommon::String const & filePath,
            bool enableRecording = true )
```

A constructor.

Parameters

filePath	a path of the MVX2 file to write to
enableRecording	an indication whether the recording shall be enabled right away

5.7.3 Member Function Documentation

5.7.3.1 EnableRecording()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileWriterGraphNode::EnableRecording (
            bool enable = true ) const
```

Enables/disables actual recording to the MVX2 file.

Parameters

enable	true in order to enable recording, false in order to disable it

5.7.3.2 SetFilePath()

```
MVX2BASICIO_API void Mvx2BasicIO::Mvx2FileWriterGraphNode::SetFilePath (
            MVCommon::String const & filePath ) const
```

Changes the path of the MVX2 file to write to.

Parameters

filePath	a new path of the MVX2 file

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/graphnodes/Mvx2FileWriterGraphNode.h

5.8 Mvx2BasicIO::NetworkReceiverGraphNode Class Reference

A source graph node for reception of frames via network.

```
#include <NetworkReceiverGraphNode.h>
```

Inherits SingleFilterGraphNode.

Public Member Functions

- MVX2BASICIO_API NetworkReceiverGraphNode (MVCommon::String const &commandsSocket, MVCommon::String const &dataSocket, uint32_t receiveBufferCapacity=5, int64_t responseReceiveTimeout=3000, bool enableIPv6=false)

  A constructor.
- MVX2BASICIO_API NetworkReceiverGraphNode (MVCommon::String const &unsupportedTransmitterProtocolVersions, MVCommon::String const &commandsSocket, MVCommon::String const &dataSocket, uint32_t receiveBufferCapacity=5, int64_t responseReceiveTimeout=3000, bool enableIPv6=false)

  A constructor.
- virtual MVX2BASICIO_API ~NetworkReceiverGraphNode ()

  A destructor.
- MVX2BASICIO_API void SetUnsupportedTransmitterProtocolVersions (MVCommon::String const &unsupportedTransmitterProtocolVersions) const

  Changes the specification of which protocol version transmitters the receiver shall not respond to.
- MVX2BASICIO_API void SetSockets (MVCommon::String const &commandsSocket, MVCommon::String const &dataSocket, bool enableIPv6=false) const

  Changes the sockets of the receiver.

5.8.1 Detailed Description

A source graph node for reception of frames via network.

Internally maintains a single receiving filter. The same filter is reused even when the graph node is added to multiple graphs.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 NetworkReceiverGraphNode() [1/2]

```
MVX2BASICIO_API Mvx2BasicIO::NetworkReceiverGraphNode::NetworkReceiverGraphNode (
            MVCommon::String const & commandsSocket,
            MVCommon::String const & dataSocket,
            uint32_t receiveBufferCapacity = 5,
            int64_t responseReceiveTimeout = 3000,
            bool enableIPv6 = false )
```

A constructor.

Parameters

commandsSocket	a socket for communication with transmitter (e.g. 'tcp://192.168.1.1:5555')
dataSocket	a socket for data reception (e.g. 'tcp://192.168.1.1:5556')
receiveBufferCapacity	a capacity of the underlying socket's receive-buffer
responseReceiveTimeout	an interval to wait for response from transmitter until the connection is considered unavailable (in ms)
enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

5.8.2.2 NetworkReceiverGraphNode() [2/2]

```
MVX2BASICIO_API Mvx2BasicIO::NetworkReceiverGraphNode::NetworkReceiverGraphNode (
            MVCommon::String const & unsupportedTransmitterProtocolVersions,
            MVCommon::String const & commandsSocket,
            MVCommon::String const & dataSocket,
            uint32_t receiveBufferCapacity = 5,
            int64_t responseReceiveTimeout = 3000,
            bool enableIPv6 = false )
```

A constructor.

Parameters

unsupportedTransmitterProtocolVersions	a comma-separated string for specifying which protocol version transmitters the receiver shall not respond to
commandsSocket	a socket for communication with transmitter (e.g. 'tcp://192.168.1.1:5555')
dataSocket	a socket for data reception (e.g. 'tcp://192.168.1.1:5556')
receiveBufferCapacity	a capacity of the underlying socket's receive-buffer
responseReceiveTimeout	an interval to wait for response from transmitter until the connection is considered unavailable (in ms)
enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

5.8.3 Member Function Documentation

5.8.3.1 SetSockets()

```
MVX2BASICIO_API void Mvx2BasicIO::NetworkReceiverGraphNode::SetSockets (
            MVCommon::String const & commandsSocket,
            MVCommon::String const & dataSocket,
            bool enableIPv6 = false ) const
```

Changes the sockets of the receiver.

Parameters

commandsSocket	a new socket for communication with transmitter (e.g. 'tcp://192.168.1.1:5555')
dataSocket	a new socket for data reception (e.g. 'tcp://192.168.1.1:5556')
enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

5.8.3.2 SetUnsupportedTransmitterProtocolVersions()

```
MVX2BASICIO_API void Mvx2BasicIO::NetworkReceiverGraphNode::SetUnsupportedTransmitterProtocol←
Versions (
            MVCommon::String const & unsupportedTransmitterProtocolVersions ) const
```

Changes the specification of which protocol version transmitters the receiver shall not respond to.

Parameters

unsupportedTransmitterProtocolVersions	a comma-separated string of protocol versions, or empty string if all versions transmitters can be responded to

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/graphnodes/NetworkReceiverGraphNode.h

5.9 Mvx2BasicIO::NetworkTransmitterGraphNode Class Reference

A target graph node for transmission of frames via network.

```
#include <NetworkTransmitterGraphNode.h>
```

Inherits SingleFilterGraphNode.

Public Member Functions

- MVX2BASICIO_API NetworkTransmitterGraphNode (MVCommon::String const &commandsSocket, MV←Common::String const &dataSocket, uint32_t sendBufferCapacity=2, bool enableTransmission=true, bool enableIPv6=false)

  A constructor.
- MVX2BASICIO_API NetworkTransmitterGraphNode (MVCommon::String const &unsupportedReceiver←ProtocolVersions, MVCommon::String const &commandsSocket, MVCommon::String const &dataSocket, uint32_t sendBufferCapacity=2, bool enableTransmission=true, bool enableIPv6=false)

  A constructor.
- virtual MVX2BASICIO_API ∼NetworkTransmitterGraphNode ()

  A destructor.
- MVX2BASICIO_API void SetUnsupportedReceiverProtocolVersions (MVCommon::String const &unsupported←ReceiverProtocolVersions) const

  Changes the specification of which protocol version receivers the transmitter shall not respond to.
- MVX2BASICIO_API void SetSockets (MVCommon::String const &commandsSocket, MVCommon::String const &dataSocket, bool enableIPv6=false) const

  Changes the sockets of the transmitter.
- MVX2BASICIO_API void EnableTransmission (bool enable=true) const

  Enables/disables actual frames transmission.
- MVX2BASICIO_API uint64_t GetDroppedAtomsCount () const

  Returns a count of dropped (not transmitted) atoms.
- MVX2BASICIO_API void ResetDroppedAtomsCounter () const

  Resets the internal counter of dropped (not transmitted) atoms.

5.9.1 Detailed Description

A target graph node for transmission of frames via network.

Internally maintains a single transmitting filter. The same filter is reused even when the graph node is added to multiple graphs.

Supports counting of dropped atoms.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 NetworkTransmitterGraphNode() [1/2]

```
MVX2BASICIO_API Mvx2BasicIO::NetworkTransmitterGraphNode::NetworkTransmitterGraphNode (
            MVCommon::String const & commandsSocket,
            MVCommon::String const & dataSocket,
            uint32_t sendBufferCapacity = 2,
            bool enableTransmission = true,
            bool enableIPv6 = false )
```

A constructor.

Parameters

commandsSocket	a socket for communication with receivers (e.g. 'tcp://192.168.1.1:5555')
dataSocket	a socket for data transmission (e.g. 'tcp://192.168.1.1:5556')
sendBufferCapacity	a capacity of the underlying socket's send-buffer
enableTransmission	an indication whether the transmission shall be enabled right away
enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

5.9.2.2 NetworkTransmitterGraphNode() [2/2]

```
MVX2BASICIO_API Mvx2BasicIO::NetworkTransmitterGraphNode::NetworkTransmitterGraphNode (
            MVCommon::String const & unsupportedReceiverProtocolVersions,
            MVCommon::String const & commandsSocket,
            MVCommon::String const & dataSocket,
            uint32_t sendBufferCapacity = 2,
            bool enableTransmission = true,
            bool enableIPv6 = false )
```

A constructor.

Parameters

unsupportedReceiverProtocolVersions	a comma-separated string for specifying which protocol version receivers the transmitter shall not respond to
commandsSocket	a socket for communication with receivers (e.g. 'tcp://192.168.1.1:5555')
dataSocket	a socket for data transmission (e.g. 'tcp://192.168.1.1:5556')
sendBufferCapacity	a capacity of the underlying socket's send-buffer
enableTransmission	an indication whether the transmission shall be enabled right away
enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

5.9.3 Member Function Documentation

5.9.3.1 EnableTransmission()

```
MVX2BASICIO_API void Mvx2BasicIO::NetworkTransmitterGraphNode::EnableTransmission (
             bool enable = true ) const
```

Enables/disables actual frames transmission.

Parameters

enable	true in order to enable transmission, false in order to disable it

5.9.3.2 GetDroppedAtomsCount()

```
MVX2BASICIO_API uint64_t Mvx2BasicIO::NetworkTransmitterGraphNode::GetDroppedAtomsCount ( )
const
```

Returns a count of dropped (not transmitted) atoms.

Returns

the count of dropped atoms

5.9.3.3 SetSockets()

```
MVX2BASICIO_API void Mvx2BasicIO::NetworkTransmitterGraphNode::SetSockets (
             MVCommon::String const & commandsSocket,
             MVCommon::String const & dataSocket,
             bool enableIPv6 = false ) const
```

Changes the sockets of the transmitter.

Parameters

commandsSocket	a new socket for communication with receivers (e.g. 'tcp://192.168.1.1:5555')
dataSocket	a new socket for data transmission (e.g. 'tcp://192.168.1.1:5556')
enableIPv6	enables IPv6 support - unless enabled, only IPv4 communication will work

5.9.3.4 SetUnsupportedReceiverProtocolVersions()

```
MVX2BASICIO_API void Mvx2BasicIO::NetworkTransmitterGraphNode::SetUnsupportedReceiverProtocol↩
Versions (
            MVCommon::String const & unsupportedReceiverProtocolVersions ) const
```

Changes the specification of which protocol version receivers the transmitter shall not respond to.

Parameters

unsupportedReceiverProtocolVersions	a comma-separated string of protocol versions, or empty string if all versions receivers can be responded to

The documentation for this class was generated from the following file:

- public/Mvx2BasicIO/graphnodes/NetworkTransmitterGraphNode.h

Index