MVCommonNet

Generated by Doxygen 1.8.16

Chapter 1

Mantis Vision: MVCommonNet

A .NET wrapper for MVCommon utilities and services.

Table of Contents

Chapter 2

Release Notes

1.2.0

Initial version (as extracted from Mvx2 framework)

Module

- **1.2.0_M1** | introduced versioning to MVCommon libraries (in the form of MVCommon::VersionInfo class and MVCommonVersion.h file)
- **1.2.0_M2** | added CUtil.h file with C preprocessor utility macros

Build support

- **1.2.0_BS1** | added MVCommon's own MVCommonConfig.cmake file for cmake support
- **1.2.0_BS2** | added MVCommonNet's own MVCommonNetConfig.cmake and MVCommonNet_iOSConfig.↩ cmake files for cmake support

Documentation

- **1.2.0_D1** | added MVCommon's own 'release notes' section to its documentation
- **1.2.0_D2** | switched documentation from xml-style comments to doxygen-style comments

2.0.0

Module

- **2.0.0_M1** | made default constructor and destructor of `NonAssignable` class protected, as there shall not exist objects of `NonAssignable` class itself
- **2.0.0_M2** | updated `libjpeg-turbo` 3rdparty dependency to version 2.0.2

Build support

- **2.0.0_BS1** | Android and LuminOS libraries size 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** | introduced a protected MVCommon::NativeObjectHolder::m_nativeObjectLock field into the MVCommon::NativeObjectHolder instances in MVCommonNet to allow its derivatives to lock the held native object during asynchronous operations on them
- **3.0.0_M2** | fixed a bug of MVCommon::NetLoggerSink in MVCommonNet which prevented parallel logging via .Net sinks
- **3.0.0_M3** | fixed MVCommon.BlockingCounter.WaitUntilValue() and MVCommon.BlockingCounter.WaitUntilValueFor() which could miss a target counter value in case the counter's value was repeatedly updated too fast, and thus not ending the waiting
- **3.0.0_M4** | extended MVCommon.BlockingCounter with a support for blocking the execution until an arbitrary condition on the counter's value is passed:
  - introduced interface MVCommon.IBlockingCounterCondition and its derivative MVCommon.BlockingCounterValueEquals,
  - introduced MVCommon.BlockingCounter.WaitUntil() and MVCommon.BlockingCounter.WaitUntilFor() functions,
  - introduced MVCommon::NetBlockingCounterCondition into MVCommonNet
- **3.0.0_M5** | refactored MVCommon.ThreadPool :
  - moved and renamed the MVCommon/legacy/MVCommon/concurrency/ThreadPool.hpp header file to MVCommon/utils/threadpool/ThreadPool.h,
  - replaced the MVCommon::ThreadPool::Job signature of jobs by an MVCommon.IThreadPoolJob interface,
  - introduced an MVCommon.ThreadPool.WaitForAnUnoccupiedThread() function,
  - refactored the implementation to support the new features,
  - introduced a .Net version of MVCommon::ThreadPool into MVCommonNet,
  - added a documentation section for the API
- **3.0.0_M6** | fixed `MVCommon::FileHelper::OpenFileReadOnly()` and `MVCommon::FileHelper::OpenFileForWriting()` utility functions which prevented concurrent reading from a file that is already open for writing or reading (affects only windows)

Build support

- **3.0.0_BS1** | CMake minimal required version increased from 3.9 to 3.14
  - updated `MVCommonConfig.cmake`, `MVCommonNetConfig.cmake` and `MVCommonNet_iOSConfig.cmake` scripts and their dependencies

4.0.0

Module

- **4.0.0_M1** | upgraded multiple internal dependencies with possible effect on:
  - MVCommon::GuidAliasDatabase
  - MVCommon::Logger and MVCommon::ILoggerSink
  - MVCommon::FileHelper

Build support

- **4.0.0_BS1** | from now on the windows libraries are compiled using msvc compiler version 142 (VS 2019)
- **4.0.0_BS2** | upgraded `cmake/toolchains/ios.cmake` toolchain file used for building for iOS platform

Documentation

- **4.0.0_D1** | introduced PDF documentation as an alternative to the HTML one:
  - `doc/MVCommon.pdf`
  - `doc/MVCommonNet.pdf`

Chapter 3

Namespace Index

3.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

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

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 6

Namespace Documentation

6.1 MVCommon Namespace Reference

Classes

- class AndroidSystemLoggerSink

  A logger sink implementation for logging log messages via Android system logging facility.
- class AppleSystemLoggerSink

  A logger sink implementation for logging log messages via Apple system logging facility.
- class BlockingCounter

  A counter with a feature of blocking a thread until the counter has a specific value.
- class BlockingCounterValueEquals

  A counter condition for checking equality of its value with a target value.
- class ByteArray

  An array of bytes.
- class CameraParams

  A data structure containing intrinsic and extrinsic parameters of cameras.
- class Color

  An RGBA color.
- class FileLoggerSink

  A logger sink implementation for logging into a file.
- class Guid

  A globally-unique identifier implementation.
- class GuidAliasDatabase

  A database of Guid aliases.
- class GuidAliasDatabaseEnumerator

  An iterator of elements of GuidAliasDatabase collections.
- class GuidGenerator

  A generator of Guids.
- class IBlockingCounterCondition

  An interface of conditions usable with blocking counters.
- class ILoggerSink

  An interface of logger sinks.
- class IThreadPoolJob

  An interface of thread pool jobs.
- class LogEntry

A log entry data structure.

- class Logger

  A logger.

- class LoggerRegistry

  A global registry of loggers.

- class Math

  A utility class for math operations.

- class Matrix4x4d

  A 4x4 matrix with double-precision floating-point values.

- class Matrix4x4f

  A 4x4 matrix with single-precision floating-point values.

- class NativeObjectHolder

  A holder of a native object with support for proper object destruction.

- class NetBlockingCounterCondition

  An abstract blocking counter condition base class intended for creation of new condition implementations in .Net environment.

- class NetLoggerSink

  An abstract logger sink base class intended for creation of new logger sink implementations in .Net environment.

- class NetThreadPoolJob

  An abstract thread pool job base class intended for creation of new job implementations in .Net environment.

- class RedirectingLoggerSink

  A logger sink implementation for redirecting log messages to another logger.

- class SharedRef

  A smart reference implementation with object reference counting.

- class **SharedRefCounter**

  A class holding count of a shared object references.

- class StdOutLoggerSink

  A logger sink implementation for logging into a standard output.

- class String

  A string implementation.

- class ThreadPool

  A pool of threads.

- class Vector2d

  A 2-dimensional vector with double-precision floating-point values.

- class Vector2f

  A 2-dimensional vector with single-precision floating-point values.

- class Vector3d

  A 3-dimensional vector with double-precision floating-point values.

- class Vector3f

  A 3-dimensional vector with single-precision floating-point values.

- class Vector4d

  A 4-dimensional vector with double-precision floating-point values.

- class Vector4f

  A 4-dimensional vector with single-precision floating-point values.

- class VersionInfo

  A structure holding module version information.

- class Versord

  A rotational quaternion (i.e. versor) with double-precision floating-point values.

- class Versorf

  A rotational quaternion (i.e. versor) with single-precision floating-point values.

Enumerations

- enum LoggerLogLevel {
  LoggerLogLevel.LLL_SILENT = 0, LoggerLogLevel.LLL_CRITICAL = LogLevel.LL_CRITICAL, LoggerLogLevel.LLL_ERROR = LogLevel.LL_ERROR, LoggerLogLevel.LLL_WARNING = LogLevel.LL_WARNING,
  LoggerLogLevel.LLL_INFO = LogLevel.LL_INFO, LoggerLogLevel.LLL_DEBUG = LogLevel.LL_DEBUG,
  LoggerLogLevel.LLL_VERBOSE = LogLevel.LL_VERBOSE }

  An enumeration of logger log levels for filtering log messages.
- enum LogLevel {
  LogLevel.LL_CRITICAL = 1, LogLevel.LL_ERROR = 2, LogLevel.LL_WARNING = 3, LogLevel.LL_INFO = 4,
  LogLevel.LL_DEBUG = 5, LogLevel.LL_VERBOSE = 6 }

  An enumeration of log levels.

6.1.1 Enumeration Type Documentation

6.1.1.1 LoggerLogLevel

`enum MVCommon.LoggerLogLevel [strong]`

An enumeration of logger log levels for filtering log messages.

Only log messages that are classified with higher log level than the logger is set to are actually logged.

Enumerator

LLL_SILENT	No log messages are logged.
LLL_CRITICAL	Only critical log messages are logged.
LLL_ERROR	Only error or higher level log messages are logged.
LLL_WARNING	Only warning and higher level log messages are logged.
LLL_INFO	Only info and higher level log messages are logged.
LLL_DEBUG	Only debug and higher level log messages are logged.
LLL_VERBOSE	All log messages are logged.

6.1.1.2 LogLevel

`enum MVCommon.LogLevel [strong]`

An enumeration of log levels.

Enumerator

LL_CRITICAL	A critical message log level.
LL_ERROR	An error message log level.
LL_WARNING	A warning message log level.
LL_INFO	An info message log level.
LL_DEBUG	A debug message log level.
LL_VERBOSE	A verbose message log level.

Chapter 7

Class Documentation

7.1 MVCommon.AndroidSystemLoggerSink Class Reference

A logger sink implementation for logging log messages via Android system logging facility.

Inherits MVCommon.ILoggerSink.

Public Member Functions

- AndroidSystemLoggerSink (LoggerLogLevel logLevel=LoggerLogLevel.LLL_VERBOSE)
  A constructor.

Additional Inherited Members

7.1.1 Detailed Description

A logger sink implementation for logging log messages via Android system logging facility.

In case the sink is instantiated on a non-Android platform, log messages are not handled at all.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 AndroidSystemLoggerSink()

```
MVCommon.AndroidSystemLoggerSink.AndroidSystemLoggerSink (
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_VERBOSE )
```

A constructor.

Parameters

logLevel	an initial log level (default value -> all log messages are processed)

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

- public/logger/sinks/AndroidSystemLoggerSink.cs

7.2 MVCommon.AppleSystemLoggerSink Class Reference

A logger sink implementation for logging log messages via Apple system logging facility.

Inherits MVCommon.ILoggerSink.

Public Member Functions

- AppleSystemLoggerSink (LoggerLogLevel logLevel=LoggerLogLevel.LLL_VERBOSE)
  A constructor.

Additional Inherited Members

7.2.1 Detailed Description

A logger sink implementation for logging log messages via Apple system logging facility.

In case the sink is instantiated on a non-Apple platform (MacOS, iOS, ...), log messages are not handled at all.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 AppleSystemLoggerSink()

```
MVCommon.AppleSystemLoggerSink.AppleSystemLoggerSink (
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_VERBOSE )
```

A constructor.

Parameters

logLevel	an initial log level (default value -> all log messages are processed)

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

- public/logger/sinks/AppleSystemLoggerSink.cs

7.3 MVCommon.BlockingCounter Class Reference

A counter with a feature of blocking a thread until the counter has a specific value.

Inherits MVCommon.NativeObjectHolder.

Public Member Functions

- BlockingCounter (Int32 initialValue=0, Int32 waitersCountHint=1)

  A constructor.
- void Increment (Int32 change=1)

  Increments the counter by a given value.
- Int32 WaitUntilValue (Int32 targetValue)

  Blocks current thread until the counter reaches given value.
- Int32 WaitUntilValueFor (Int32 targetValue, UInt64 milliseconds)

  Blocks current thread until the counter reaches given value or until a timeout expires.
- Int32 WaitUntil (IBlockingCounterCondition condition)

  Blocks current thread until the counter's value is accepted by a condition.
- Int32 WaitUntilFor (IBlockingCounterCondition condition, UInt64 milliseconds)

  Blocks current thread until the counter's value is accepted by a condition or until a timeout expires.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- Int32 Value `[get]`

  A getter of the current counter's value.

Additional Inherited Members

7.3.1 Detailed Description

A counter with a feature of blocking a thread until the counter has a specific value.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 BlockingCounter()

```
MVCommon.BlockingCounter.BlockingCounter (
            Int32 initialValue = 0,
            Int32 waitersCountHint = 1 )
```

A constructor.

Parameters

initialValue	an initial value of the counter
waitersCountHint	a hint about expected count of waiting threads - it allows an optimization of internal memory allocations made per each waiting call in cases when count of parallel waiters can be predicted. Special value `0` will result in allocations made every time, and `negative` hint value results in no deallocations (and thus maximum reusability of the memory) during the entire lifetime of the counter.

7.3.3 Member Function Documentation

7.3.3.1 Increment()

```
void MVCommon.BlockingCounter.Increment (
            Int32 change = 1 )
```

Increments the counter by a given value.

Parameters

change	a change to increase the counter's value by (may be negative)

7.3.3.2 WaitUntil()

```
Int32 MVCommon.BlockingCounter.WaitUntil (
            IBlockingCounterCondition condition )
```

Blocks current thread until the counter's value is accepted by a condition.

Parameters

condition	a condition that must pass in order to unblock the thread

Returns

the value which was accepted by the condition

7.3.3.3 WaitUntilFor()

```
Int32 MVCommon.BlockingCounter.WaitUntilFor (
            IBlockingCounterCondition condition,
            UInt64 milliseconds )
```

Blocks current thread until the counter's value is accepted by a condition or until a timeout expires.

Parameters

condition	a condition that must pass in order to unblock the thread
milliseconds	a timeout (in milliseconds) after which the current thread is unblocked at the latest

Returns

the value accepted by the condition when the counter reaches it before the timeout expires, current counter's value otherwise

7.3.3.4 WaitUntilValue()

```
Int32 MVCommon.BlockingCounter.WaitUntilValue (
            Int32 targetValue )
```

Blocks current thread until the counter reaches given value.

Parameters

targetValue	a value the counter has to reach in order to unblock the thread

Returns

the target value

7.3.3.5 WaitUntilValueFor()

```
Int32 MVCommon.BlockingCounter.WaitUntilValueFor (
            Int32 targetValue,
            UInt64 milliseconds )
```

Blocks current thread until the counter reaches given value or until a timeout expires.

Parameters

targetValue	a value the counter has to reach in order to unblock the thread
milliseconds	a timeout (in milliseconds) after which the current thread is unblocked at the latest

Returns

the target value when the counter reaches it before the timeout expires, current counter's value otherwise

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

- public/utils/blockingcounter/BlockingCounter.cs

7.4 MVCommon.BlockingCounterValueEquals Class Reference

A counter condition for checking equality of its value with a target value.

Inherits MVCommon.IBlockingCounterCondition.

Public Member Functions

- BlockingCounterValueEquals (Int32 targetValue)
  A constructor.

Additional Inherited Members

7.4.1 Detailed Description

A counter condition for checking equality of its value with a target value.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 BlockingCounterValueEquals()

```
MVCommon.BlockingCounterValueEquals.BlockingCounterValueEquals (
            Int32 targetValue )
```

A constructor.

Parameters

targetValue	a target value

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

- public/utils/blockingcounter/BlockingCounterValueEquals.cs

7.5 MVCommon.ByteArray Class Reference

An array of bytes.

Inherits MVCommon.NativeObjectHolder, and IEquatable< ByteArray >.

Public Member Functions

- ByteArray ()

  A constructor.
- ByteArray (byte[ ] data)

  A constructor.
- ByteArray (byte aByte, UInt64 count=1)

  A constructor.
- ByteArray (IntPtr nativeObject)

  A constructor.
- void Clear ()

  Empties the array.
- ByteArray Push (ByteArray other)

  Pushes another array of bytes to the end of this array.
- ByteArray Push (byte aByte, UInt64 count=1)

  Pushes an array of the same byte to the end of this array.
- ByteArray Push (byte[ ] data)

  Pushes data to the end of this array.
- byte Pop ()

  Pops and removes a single byte from the front of the array.
- ByteArray Pop (UInt64 count)

  Pops an array of bytes from the front of the array.
- void Skip (UInt64 count=1)

  Skips an array of bytes from the front of the array.
- ByteArray Subarray (UInt64 startPos=0, UInt64 count=1)

  Creates a subarray of bytes from the array, not removing the bytes from the original array.
- ByteArray Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static ByteArray operator+ (ByteArray lhs, ByteArray rhs)

  Pushes an array of bytes to the end of another array.
- static ByteArray operator+ (ByteArray lhs, byte aByte)

  Pushes a byte to the end of an array.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeByteArrayObject `[get]`

  A getter of the native ByteArray object.
- IntPtr NativeDataPtr `[get]`

  Gets a pointer to the native array's internal continuous memory.
- byte[ ] NetArray `[get]`

  Constructs a .Net array of bytes from the full content of the array.
- UInt64 Size `[get]`

  Gets size of the array.
- byte this[UInt64 i] `[get, set]`

  A property for accessing specific array element (byte).

Additional Inherited Members

7.5.1 Detailed Description

An array of bytes.

The implementation maintains a continuous array (vector) of bytes (uint8_t), which is resized when necessary and under specific conditions for maximum efficiency. The array provides operations for pushing bytes to the end of the array and for popping them from the array's front, behaving thus like a queue. The difference from std::queue is that the array's internal storage is continuous.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 ByteArray() [1/4]

```
MVCommon.ByteArray.ByteArray ( )
```

A constructor.

Creates an empty array of bytes.

7.5.2.2 ByteArray() [2/4]

```
MVCommon.ByteArray.ByteArray (
            byte[] data )
```

A constructor.

Parameters

data	a data to initialize the array with

Creates an array of bytes initialized with the given data.

7.5.2.3 ByteArray() [3/4]

```
MVCommon.ByteArray.ByteArray (
            byte aByte,
            UInt64 count = 1 )
```

A constructor.

Parameters

aByte	a byte to initialize the array with
count	a count of bytes to initialize the array with

Creates an array of bytes containing the given amount of the same byte.

7.5.2.4 ByteArray() [4/4]

```
MVCommon.ByteArray.ByteArray (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native ByteArray object

7.5.3 Member Function Documentation

7.5.3.1 Clone()

```
ByteArray MVCommon.ByteArray.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.5.3.2 operator+() [1/2]

```
static ByteArray MVCommon.ByteArray.operator+ (
            ByteArray lhs,
            byte aByte )  [static]
```

Pushes a byte to the end of an array.

Parameters

lhs	an array to push into
aByte	a byte to push

Returns

the array that was pushed into

7.5.3.3 operator+() [2/2]

```
static ByteArray MVCommon.ByteArray.operator+ (
            ByteArray lhs,
            ByteArray rhs )  [static]
```

Pushes an array of bytes to the end of another array.

Parameters

lhs	an array to push into
rhs	an array to push

Returns

the array that was pushed into

7.5.3.4 Pop() [1/2]

```
byte MVCommon.ByteArray.Pop ( )
```

Pops and removes a single byte from the front of the array.

Returns

the front byte

Exceptions

System.InvalidOperationException	raised when there are no data available in the array

7.5.3.5 Pop() [2/2]

```
ByteArray MVCommon.ByteArray.Pop (
            UInt64 count )
```

Pops an array of bytes from the front of the array.

Parameters

count	a count of bytes to pop

Returns

the array of bytes popped from the front

The call always succeeds, even when there is not enough bytes in the array. The returned array will in such case contain less bytes than requested.

7.5.3.6 Push() [1/3]

```
ByteArray MVCommon.ByteArray.Push (
            byte aByte,
            UInt64 count = 1 )
```

Pushes an array of the same byte to the end of this array.

Parameters

aByte	a byte to push
count	a count of bytes to push

Returns

this array

7.5.3.7 Push() [2/3]

```
ByteArray MVCommon.ByteArray.Push (
            byte[] data )
```

Pushes data to the end of this array.

Parameters

data	a data to push

Returns

this array

7.5.3.8 Push() [3/3]

```
ByteArray MVCommon.ByteArray.Push (
            ByteArray other )
```

Pushes another array of bytes to the end of this array.

Parameters

other	an array to push to this array

Returns

this array

7.5.3.9 Skip()

```
void MVCommon.ByteArray.Skip (
            UInt64 count = 1 )
```

Skips an array of bytes from the front of the array.

Parameters

count	a count of bytes to skip

The call always succeeds, even when there is not enough bytes in the array.

7.5.3.10 Subarray()

```
ByteArray MVCommon.ByteArray.Subarray (
            UInt64 startPos = 0,
            UInt64 count = 1 )
```

Creates a subarray of bytes from the array, not removing the bytes from the original array.

Parameters

startPos	a position of the first byte
count	a count of bytes

Returns

the subarray of bytes

The call always succeeds, even when there is not enough bytes in the original array or when the starting position is outside of the valid range. The returned array will in such case contain less bytes than requested or even no bytes at all.

7.5.4 Property Documentation

7.5.4.1 NativeDataPtr

```
IntPtr MVCommon.ByteArray.NativeDataPtr  [get]
```

Gets a pointer to the native array's internal continuous memory.

Returns

a pointer to the native array's memory

The call may return different pointers at different times, for example when some bytes were already popped. The returned pointer always points to the next byte that would be popped if such a call took place.

7.5.4.2 NetArray

```
byte [] MVCommon.ByteArray.NetArray  [get]
```

Constructs a .Net array of bytes from the full content of the array.

The returned .Net array is independent from this array, so no modifications are reflected back.

Because of the limitations of .Net, only a subarray of bytes can be constructed with maximum length of Int32.↩ MaxValue elements.

Returns

a .Net array of bytes

7.5.4.3 Size

```
UInt64 MVCommon.ByteArray.Size  [get]
```

Gets size of the array.

Returns

array's size

7.5.4.4 this[UInt64 i]

```
byte MVCommon.ByteArray.this[UInt64 i]  [get], [set]
```

A property for accessing specific array element (byte).

Parameters

i	byte's index

Returns

a byte at the index

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

- public/utils/ByteArray.cs

7.6 MVCommon.CameraParams Class Reference

A data structure containing intrinsic and extrinsic parameters of cameras.

Inherits MVCommon.NativeObjectHolder, and IEquatable< CameraParams >.

Public Member Functions

- float GetDistortionCoefficient (UInt32 index)
  Returns a distortion coefficient.
- void SetDistortionCoefficient (UInt32 index, float coefficient)
  Sets a distortion coefficient.
- CameraParams ()
  A constructor.
- CameraParams (UInt32 width, UInt32 height)
  A constructor.
- CameraParams (UInt32 width, UInt32 height, Vector2f F)
  A constructor.
- CameraParams (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the camera params into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the camera params into a byte array.
- void NormalizePoint (Vector2f point)
  Normalizes a point coordinates using focal lengths and principal point offsets of camera.
- void NormalizePoint (Vector3f point)
  Normalizes a point x and y coordinates using focal lengths and principal point offsets of camera.
- void DenormalizePoint (Vector2f point)
  Denormalizes a point coordinates using focal lengths and principal point offsets of camera.
- void DenormalizePoint (Vector3f point)
  Denormalizes a point x and y coordinates using focal lengths and principal point offsets of camera.
- void UndistortPoint (Vector2f point)
  Transforms a point coordinates to compensate camera lens distortion.
- void UndistortPoint (Vector3f point)
  Transforms a point x and y coordinates to compensate camera lens distortion.
- CameraParams ScaleToResolution (UInt32 targetWidth, UInt32 targetHeight)
  Creates a new camera params with focal lengths and principal point offsets scaled for a target resolution.
- CameraParams Clone ()
  Makes an independent clone object.

Static Public Member Functions

- static CameraParams FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes camera params from a byte array.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = CameraParams_GetRawBytesSize()

  A constant indicating the size of raw bytes of the camera params.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- UInt32 width `[get, set]`

  A width (in pixels).
- UInt32 height `[get, set]`

  A height (in pixels).
- Vector2f F `[get, set]`

  Focal lengths (in pixels) in x and y directions.
- Vector2f C `[get, set]`

  Principal point offsets (in pixels) in x and y directions.
- Vector2f distortionC `[get, set]`

  Principal point offsets for distortion operations (in pixels) in x and y directions.
- Vector3f translation `[get, set]`

  Translation offset of the origin in the camera's coordinate system.
- Matrix4x4f rotation `[get, set]`

  Rotation matrix offset of the origin in the camera's coordinate system.
- IntPtr nativeCameraParamsObject `[get]`

  A getter of the native CameraParams object.

Additional Inherited Members

7.6.1 Detailed Description

A data structure containing intrinsic and extrinsic parameters of cameras.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 CameraParams() [1/4]

```
MVCommon.CameraParams.CameraParams ( )
```

A constructor.

Constructs the camera params with default values - width and height equal to 0, principal point offsets equal to [0, 0], focal lengths equal to [1, 1], and distortion, translation and rotation set to identities.

7.6.2.2 CameraParams() [2/4]

```
MVCommon.CameraParams.CameraParams (
            UInt32 width,
            UInt32 height )
```

A constructor.

Parameters

width	a width (in pixels)
height	a height (in pixels)

Constructs the camera params with default values - principal point offsets equal to [width/2, height/2], focal lengths equal to [1, 1], and distortion, translation and rotation set to identities.

7.6.2.3 CameraParams() [3/4]

```
MVCommon.CameraParams.CameraParams (
            UInt32 width,
            UInt32 height,
            Vector2f F )
```

A constructor.

Parameters

width	a width (in pixels)
height	a height (in pixels)
F	focal lengths (in pixels) in x and y directions

Constructs the camera params with default values - principal point offsets equal to [width/2, height/2], and distortion, translation and rotation set to identities.

7.6.2.4 CameraParams() [4/4]

```
MVCommon.CameraParams.CameraParams (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native CameraParams object

7.6.3 Member Function Documentation

7.6.3.1 Clone()

```
CameraParams MVCommon.CameraParams.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.6.3.2 DenormalizePoint() [1/2]

```
void MVCommon.CameraParams.DenormalizePoint (
            Vector2f point )
```

Denormalizes a point coordinates using focal lengths and principal point offsets of camera.

Parameters

point	a point to denormalize

7.6.3.3 DenormalizePoint() [2/2]

```
void MVCommon.CameraParams.DenormalizePoint (
            Vector3f point )
```

Denormalizes a point x and y coordinates using focal lengths and principal point offsets of camera.

Parameters

point	a point to denormalize

7.6.3.4 FromRawBytes()

```
static CameraParams MVCommon.CameraParams.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes camera params from a byte array.

Parameters

bytes	an array of camera params bytes
consumeBytes	determines whether bytes of the camera params shall be removed from the array

Returns

camera params

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.6.3.5 GetDistortionCoefficient()

```
float MVCommon.CameraParams.GetDistortionCoefficient (
            UInt32 index )
```

Returns a distortion coefficient.

Parameters

index	an index of the distortion coefficient (from range [0, 4])

Returns

the coefficient

7.6.3.6 NormalizePoint() [1/2]

```
void MVCommon.CameraParams.NormalizePoint (
            Vector2f point )
```

Normalizes a point coordinates using focal lengths and principal point offsets of camera.

Parameters

point	a point to normalize

7.6.3.7 NormalizePoint() [2/2]

```
void MVCommon.CameraParams.NormalizePoint (
            Vector3f point )
```

Normalizes a point x and y coordinates using focal lengths and principal point offsets of camera.

Parameters

point	a point to normalize

7.6.3.8 ScaleToResolution()

```
CameraParams MVCommon.CameraParams.ScaleToResolution (
            UInt32 targetWidth,
            UInt32 targetHeight )
```

Creates a new camera params with focal lengths and principal point offsets scaled for a target resolution.

Parameters

targetWidth	target width
targetHeight	target height

Returns

a new camera params

Vertical (y) and horizontal (x) elements are scaled independently. Distortion, translation and rotation are preserved in the new camera params.

7.6.3.9 SetDistortionCoefficient()

```
void MVCommon.CameraParams.SetDistortionCoefficient (
            UInt32 index,
            float coefficient )
```

Sets a distortion coefficient.

Parameters

index	an index of the distortion coefficient (from range [0, 4])
coefficient	a new coefficient value

7.6.3.10 ToCommonString()

```
String MVCommon.CameraParams.ToCommonString ( )
```

Converts the camera params into a human-readable string.

Returns

the camera params string

7.6.3.11 ToRawBytes()

```
void MVCommon.CameraParams.ToRawBytes (
            ByteArray bytes )
```

Serializes the camera params into a byte array.

Parameters

bytes	a byte array to serialize into

7.6.3.12 UndistortPoint() [1/2]

```
void MVCommon.CameraParams.UndistortPoint (
            Vector2f point )
```

Transforms a point coordinates to compensate camera lens distortion.

Parameters

point	a point to undistort

7.6.3.13 UndistortPoint() [2/2]

```
void MVCommon.CameraParams.UndistortPoint (
            Vector3f point )
```

Transforms a point x and y coordinates to compensate camera lens distortion.

Parameters

point	a point to undistort

7.6.4 Property Documentation

7.6.4.1 distortionC

```
Vector2f MVCommon.CameraParams.distortionC  [get], [set]
```

Principal point offsets for distortion operations (in pixels) in x and y directions.

Its value may be slightly different than the value of C.

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

- public/data/CameraParams.cs

7.7 MVCommon.Color Class Reference

An RGBA color.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Color >.

Public Member Functions

- Color ()
  A constructor of the black color.
- Color (byte redByte, byte greenByte, byte blueByte, byte alphaByte=255)
  A constructor.
- Color (float red, float green, float blue, float alpha=1.0f)
  A constructor.
- Color (Vector4f color)
  A constructor.
- Color (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()

Converts the color into a human-readable string.

- String ToRGB_HTMLString ()

  Converts the RGB part of the color into a HTML hexadecimal string in format #rrggbb.

- void SetValue (byte redByte, byte greenByte, byte blueByte, byte alphaByte=255)

  Sets value of the color.

- void SetValue (float red, float green, float blue, float alpha=1.0f)

  Sets value of the color.

- void SetValue (MVCommon.Vector4f color)

  Sets value of the color.

- byte GetRGBBrightnessByte ()

  Calculates an RGB brightness byte value of the color.

- float GetRGBBrightness ()

  Calculates an RGB brightness value of the color in range <0.0, 1.0>.

- Color Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Color FromString (String str)

  Creates a color from a human-readable string.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- byte redByte `[get, set]`

  A red element byte value.

- byte greenByte `[get, set]`

  A green element byte value.

- byte blueByte `[get, set]`

  A blue element byte value.

- byte alphaByte `[get, set]`

  An alpha element byte value.

- float red `[get, set]`

  A red element value in range <0.0, 1.0>.

- float green `[get, set]`

  A green element value in range <0.0, 1.0>.

- float blue `[get, set]`

  A blue element value in range <0.0, 1.0>.

- float alpha `[get, set]`

  An alpha element value in range <0.0, 1.0>.

- IntPtr nativeColorObject `[get]`

  A getter of the native Color object.

Additional Inherited Members

7.7.1 Detailed Description

An RGBA color.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 Color() [1/4]

```
MVCommon.Color.Color (
            byte redByte,
            byte greenByte,
            byte blueByte,
            byte alphaByte = 255 )
```

A constructor.

Parameters

redByte	a red element byte value
greenByte	a green element byte value
blueByte	a blue element byte value
alphaByte	an alpha element byte value

7.7.2.2 Color() [2/4]

```
MVCommon.Color.Color (
            float red,
            float green,
            float blue,
            float alpha = 1.0f )
```

A constructor.

Parameters

red	a red element value in range <0.0, 1.0>
green	a green element value in range <0.0, 1.0>
blue	a blue element value in range <0.0, 1.0>
alpha	an alpha element value in range <0.0, 1.0>

7.7.2.3 Color() [3/4]

```
MVCommon.Color.Color (
            Vector4f color )
```

A constructor.

Parameters

color	a vector containing color element values in range <0.0, 1.0> (x -> red, y -> green, z -> blue, w -> alpha)

7.7.2.4 Color() [4/4]

```
MVCommon.Color.Color (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native Color object

7.7.3 Member Function Documentation

7.7.3.1 Clone()

```
Color MVCommon.Color.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.7.3.2 FromString()

```
static Color MVCommon.Color.FromString (
            String str )  [static]
```

Creates a color from a human-readable string.

Parameters

str	a color string

Returns

a color

7.7.3.3 GetRGBBrightness()

```
float MVCommon.Color.GetRGBBrightness ( )
```

Calculates an RGB brightness value of the color in range <0.0, 1.0>.

Returns

an RGB brightness value

7.7.3.4 GetRGBBrightnessByte()

```
byte MVCommon.Color.GetRGBBrightnessByte ( )
```

Calculates an RGB brightness byte value of the color.

Returns

an RGB brightness byte value

7.7.3.5 SetValue() [1/3]

```
void MVCommon.Color.SetValue (
            byte redByte,
            byte greenByte,
            byte blueByte,
            byte alphaByte = 255 )
```

Sets value of the color.

Parameters

redByte	a new red element byte value
greenByte	a new green element byte value
blueByte	a new blue element byte value
alphaByte	a new alpha element byte value

7.7.3.6 SetValue() [2/3]

```
void MVCommon.Color.SetValue (
            float red,
            float green,
            float blue,
            float alpha = 1.0f )
```

Sets value of the color.

Parameters

red	a new red element value in range <0.0, 1.0>
green	a new green element value in range <0.0, 1.0>
blue	a new blue element value in range <0.0, 1.0>
alpha	a new alpha element value in range <0.0, 1.0>

7.7.3.7 SetValue() [3/3]

```
void MVCommon.Color.SetValue (
            MVCommon.Vector4f color )
```

Sets value of the color.

Parameters

color	a vector containing new color element values in range <0.0, 1.0> (x -> red, y -> green, z -> blue, w -> alpha)

7.7.3.8 ToCommonString()

```
String MVCommon.Color.ToCommonString ( )
```

Converts the color into a human-readable string.

Returns

the color string

7.7.3.9 ToRGB_HTMLString()

```
String MVCommon.Color.ToRGB_HTMLString ( )
```

Converts the RGB part of the color into a HTML hexadecimal string in format #rrggbb.

Returns

the RGB HTML hexadecimal string

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

- public/data/Color.cs

7.8 MVCommon.FileLoggerSink Class Reference

A logger sink implementation for logging into a file.

Inherits MVCommon.ILoggerSink.

Public Member Functions

- FileLoggerSink (String path, LoggerLogLevel logLevel=LoggerLogLevel.LLL_VERBOSE)
  A constructor.

Additional Inherited Members

7.8.1 Detailed Description

A logger sink implementation for logging into a file.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 FileLoggerSink()

```
MVCommon.FileLoggerSink.FileLoggerSink (
            String path,
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_VERBOSE )
```

A constructor.

Parameters

path	a path of the file
logLevel	an initial log level (default value -> all log messages are processed)

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

- public/logger/sinks/FileLoggerSink.cs

7.9 MVCommon.Guid Class Reference

A globally-unique identifier implementation.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Guid >.

Public Member Functions

- Guid ()

  A constructor of a Guid with all bytes set to 0.
- Guid (IntPtr nativeObject)

  A constructor.
- String ToHexString ()

  Formats the Guid to hexadecimal 00000000-0000-0000-0000-000000000000 format.
- void ToRawBytes (ByteArray bytes)

  Formats the Guid into a raw bytes array.
- void ToRfc4122 (ByteArray bytes)

  Formats the Guid into RFC 4122 format.
- Guid Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Guid Nil ()

  Constructs a new Nil Guid (with all bytes set to 0).
- static Guid FromHexString (String str)

  Parses a string in hexadecimal format 00000000-0000-0000-0000-000000000000 into a Guid.
- static Guid FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Constructs a Guid using a raw bytes array (must contain 16 elements).
- static Guid FromRfc4122 (ByteArray bytes, bool consumeBytes=false)

  Constructs a Guid using an array of bytes in RFC 4122 format (must contain 16 elements).

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Guid_GetRawBytesSize()

  A constant indicating the size of raw bytes of the Guid.
- static readonly UInt64 RFC4122_BYTES_SIZE = Guid_GetRfc4122BytesSize()

  A constant indicating the size of bytes in RFC 4122 format of the Guid.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeGuidObject `[get]`

  A getter of the native Guid object.

Additional Inherited Members

7.9.1 Detailed Description

A globally-unique identifier implementation.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 Guid()

```
MVCommon.Guid.Guid (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native Guid object

7.9.3 Member Function Documentation

7.9.3.1 Clone()

```
Guid MVCommon.Guid.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.9.3.2 FromHexString()

```
static Guid MVCommon.Guid.FromHexString (
            String str )  [static]
```

Parses a string in hexadecimal format 00000000-0000-0000-0000-000000000000 into a Guid.

Parameters

str	a string to parse

Returns

a Guid

The input string must be at least 32 characters long (i.e. 32 hexa characters). It can optionally contain an opening and a closing bracket ('{' and '}') and 4 hyphens on specific positions of the string.

Exceptions

System.ArgumentException	raised when the format of the string is invalid and can not be parsed

7.9.3.3 FromRawBytes()

```
static Guid MVCommon.Guid.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Constructs a Guid using a raw bytes array (must contain 16 elements).

Parameters

bytes	an array of 16 bytes
consumeBytes	an indication whether the bytes of the array shall be consumed

Returns

a Guid

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.9.3.4 FromRfc4122()

```
static Guid MVCommon.Guid.FromRfc4122 (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Constructs a Guid using an array of bytes in RFC 4122 format (must contain 16 elements).

Parameters

bytes	an array of 16 bytes in RFC 4122 format
consumeBytes	an indication whether the bytes of the array shall be consumed

Returns

a Guid

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.9.3.5 Nil()

```
static Guid MVCommon.Guid.Nil ( )  [static]
```

Constructs a new Nil Guid (with all bytes set to 0).

Returns

a Nil Guid

7.9.3.6 ToHexString()

```
String MVCommon.Guid.ToHexString ( )
```

Formats the Guid to hexadecimal 00000000-0000-0000-0000-000000000000 format.

Returns

a string of 36 characters (32 for hexa characters and 4 for hyphens)

7.9.3.7 ToRawBytes()

```
void MVCommon.Guid.ToRawBytes (
            ByteArray bytes )
```

Formats the Guid into a raw bytes array.

Parameters

bytes	an array to store 16 raw bytes in

7.9.3.8 ToRfc4122()

```
void MVCommon.Guid.ToRfc4122 (
            ByteArray bytes )
```

Formats the Guid into RFC 4122 format.

Parameters

bytes	an array to store 16 raw bytes in RFC 4122 format in

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

- public/guid/Guid.cs

7.10 MVCommon.GuidAliasDatabase Class Reference

A database of Guid aliases.

Inherits MVCommon.NativeObjectHolder, IEnumerable< KeyValuePair< Guid, String >>, and IEquatable< GuidAliasDatabase >.

Public Member Functions

- GuidAliasDatabase ()
  A constructor.
- GuidAliasDatabase (IntPtr nativeObject)
  A constructor.
- void RegisterGuidAlias (Guid guid, String alias)
  Registers a new Guid alias.
- void UnregisterGuidAlias (Guid guid)
  Unregisters a Guid alias.
- void UnregisterGuidAlias (String alias)
  Unregisters a Guid alias.
- bool TryGetGuidAlias (Guid guid, String alias)
  Tries to get an alias registered for a given Guid.
- bool TryGetGuidWithAlias (String alias, Guid guid)
  Tries to get a Guid with an alias registered.
- String GetGuidAlias (Guid guid)
  Gets an alias registered for a given Guid.
- String GetGuidAlias (Guid guid, String fallbackAlias)

Gets an alias registered for a given Guid.

- Guid GetGuidWithAlias (String alias)

  Gets a Guid with an alias registered.

- Guid GetGuidWithAlias (String alias, Guid fallbackGuid)

  Gets a Guid with an alias registered.

- bool GuidRegistered (Guid guid)

  Checks whether a Guid has already an alias registered.

- bool AliasRegistered (String alias)

  Checks whether there already is a Guid with an alias registered.

- GuidAliasDatabase Clone ()

  Makes an independent clone object.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Additional Inherited Members

7.10.1 Detailed Description

A database of Guid aliases.

The database keeps pairs of Guid and String alias objects and provides fast bi-directional mapping between them. Each Guid can only have a single alias assigned and each alias can only be assigned to a single Guid.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 GuidAliasDatabase()

```
MVCommon.GuidAliasDatabase.GuidAliasDatabase (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native GuidAliasDatabase object

7.10.3 Member Function Documentation

7.10.3.1 AliasRegistered()

```
bool MVCommon.GuidAliasDatabase.AliasRegistered (
            String alias )
```

Checks whether there already is a Guid with an alias registered.

Parameters

alias	an alias to check

Returns

true in case there already is a Guid registered with the alias

7.10.3.2 Clone()

```
GuidAliasDatabase MVCommon.GuidAliasDatabase.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.10.3.3 GetGuidAlias() [1/2]

```
String MVCommon.GuidAliasDatabase.GetGuidAlias (
            Guid guid )
```

Gets an alias registered for a given Guid.

Parameters

guid	a Guid to get the alias for

Returns

an alias of the Guid or an empty string in case there is none

7.10.3.4 GetGuidAlias() [2/2]

```
String MVCommon.GuidAliasDatabase.GetGuidAlias (
            Guid guid,
            String fallbackAlias )
```

Gets an alias registered for a given Guid.

Parameters

guid	a Guid to get the alias for
fallbackAlias	a string returned in case there is no alias registered for the Guid

Returns

an alias of the Guid or the fallback string in case there is none

7.10.3.5 GetGuidWithAlias() [1/2]

```
Guid MVCommon.GuidAliasDatabase.GetGuidWithAlias (
            String alias )
```

Gets a Guid with an alias registered.

Parameters

alias	an alias to look a Guid registered with for

Returns

a Guid registered with the alias or Nil Guid in case there is none

7.10.3.6 GetGuidWithAlias() [2/2]

```
Guid MVCommon.GuidAliasDatabase.GetGuidWithAlias (
            String alias,
            Guid fallbackGuid )
```

Gets a Guid with an alias registered.

Parameters

alias	an alias to look a Guid registered with for
fallbackGuid	a Guid returned in case there is no Guid registered with the alias

Returns

a Guid registered with the alias or fallback Guid in case there is none

7.10.3.7 GuidRegistered()

```
bool MVCommon.GuidAliasDatabase.GuidRegistered (
            Guid guid )
```

Checks whether a Guid has already an alias registered.

Parameters

guid	a Guid to check

Returns

true in case the Guid already has an alias registered

7.10.3.8 RegisterGuidAlias()

```
void MVCommon.GuidAliasDatabase.RegisterGuidAlias (
            Guid guid,
            String alias )
```

Registers a new Guid alias.

Parameters

guid	a Guid to register alias for
alias	an alias of the Guid

Exceptions

System.ArgumentException	raised when there already is a different alias registered for the given Guid or the alias is already registered with another Guid

7.10.3.9 TryGetGuidAlias()

```
bool MVCommon.GuidAliasDatabase.TryGetGuidAlias (
            Guid guid,
            String alias )
```

Tries to get an alias registered for a given Guid.

Parameters

guid	a Guid to get the alias for
alias	a target to store the alias to

Returns

true in case there is an alias registered for the Guid

7.10.3.10 TryGetGuidWithAlias()

```
bool MVCommon.GuidAliasDatabase.TryGetGuidWithAlias (
            String alias,
            Guid guid )
```

Tries to get a Guid with an alias registered.

Parameters

alias	an alias to look a Guid registered with for
guid	a target to store the Guid to

Returns

true in case there is a Guid registered with the alias

7.10.3.11 UnregisterGuidAlias() [1/2]

```
void MVCommon.GuidAliasDatabase.UnregisterGuidAlias (
            Guid guid )
```

Unregisters a Guid alias.

Parameters

guid	a Guid to unregister alias of

7.10.3.12 UnregisterGuidAlias() [2/2]

```
void MVCommon.GuidAliasDatabase.UnregisterGuidAlias (
            String alias )
```

Unregisters a Guid alias.

Parameters

alias	an alias to unregister

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

- public/guid/GuidAliasDatabase.cs

7.11 MVCommon.GuidAliasDatabaseEnumerator Class Reference

An iterator of elements of GuidAliasDatabase collections.

Inherits IEnumerator< KeyValuePair< Guid, String >>.

Public Member Functions

- GuidAliasDatabaseEnumerator (GuidAliasDatabase guidAliasDatabase)
  A constructor.

7.11.1 Detailed Description

An iterator of elements of GuidAliasDatabase collections.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 GuidAliasDatabaseEnumerator()

```
MVCommon.GuidAliasDatabaseEnumerator.GuidAliasDatabaseEnumerator (
            GuidAliasDatabase guidAliasDatabase )
```

A constructor.

Parameters

guidAliasDatabase	a database to enumerate elements of

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

- public/guid/GuidAliasDatabaseEnumerator.cs

7.12 MVCommon.GuidGenerator Class Reference

A generator of Guids.

Static Public Member Functions

- static Guid GenerateGuid (Guid guidNamespace, String seed)

  Generates a Guid based on another Guid (a namespace) and a string seed.
- static Guid GenerateGuid (String seed)

  Generates a Guid based on a string seed.

7.12.1 Detailed Description

A generator of Guids.

7.12.2 Member Function Documentation

7.12.2.1 GenerateGuid() [1/2]

```
static Guid MVCommon.GuidGenerator.GenerateGuid (
            Guid guidNamespace,
            String seed )  [static]
```

Generates a Guid based on another Guid (a namespace) and a string seed.

Using the same Guid namespace and the same seed will always produce the same generated Guid.

Parameters

guidNamespace	a Guid in the role of a namespace (ancestor) for the new Guid
seed	a seed for the new Guid generation

Returns

generated Guid

7.12.2.2 GenerateGuid() [2/2]

```
static Guid MVCommon.GuidGenerator.GenerateGuid (
            String seed )  [static]
```

Generates a Guid based on a string seed.

Using the same seed will always produce the same generated Guid.

Parameters

seed	a seed for the new Guid generation

Returns

generated Guid

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

- public/guid/GuidGenerator.cs

7.13 MVCommon.IBlockingCounterCondition Class Reference

An interface of conditions usable with blocking counters.

Inherits MVCommon.NativeObjectHolder, and IEquatable< IBlockingCounterCondition >.

Inherited by MVCommon.BlockingCounterValueEquals, and MVCommon.NetBlockingCounterCondition.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.

Properties

- IntPtr nativeBlockingCounterConditionObject `[get]`
  A getter of the native IBlockingCounterCondition object.

Additional Inherited Members

7.13.1 Detailed Description

An interface of conditions usable with blocking counters.

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

- public/utils/blockingcounter/IBlockingCounterCondition.cs

7.14 MVCommon.ILoggerSink Class Reference

An interface of logger sinks.

Inherits MVCommon.NativeObjectHolder, and IEquatable< ILoggerSink >.

Inherited by MVCommon.AndroidSystemLoggerSink, MVCommon.AppleSystemLoggerSink, MVCommon.FileLoggerSink, MVCommon.NetLoggerSink, MVCommon.RedirectingLoggerSink, and MVCommon.StdOutLoggerSink.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeLoggerSinkObject `[get]`

  A getter of the native ILoggerSink object.
- LoggerLogLevel LogLevel `[get, set]`

  A property for setting and accessing log level of the logger sink for log messages filtering.

Additional Inherited Members

7.14.1 Detailed Description

An interface of logger sinks.

When a logger sink is attached to a logger, it will receive all log messages logged via that logger, assuming that the log message is not filtered by the sink's own log level setting.

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

- public/logger/ILoggerSink.cs

7.15 MVCommon.IThreadPoolJob Class Reference

An interface of thread pool jobs.

Inherits MVCommon.NativeObjectHolder, and IEquatable< IThreadPoolJob >.

Inherited by MVCommon.NetThreadPoolJob.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeThreadPoolJobObject `[get]`

  A getter of the native IThreadPoolJob object.

Additional Inherited Members

7.15.1 Detailed Description

An interface of thread pool jobs.

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

- public/utils/threadpool/IThreadPoolJob.cs

7.16 MVCommon.LogEntry Class Reference

A log entry data structure.

Inherits MVCommon.NativeObjectHolder.

Public Member Functions

- LogEntry Clone ()
  Makes an independent clone object.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.

Properties

- LogLevel Level `[get]`
  A getter of the log entry's level.
- String Tag `[get]`
  A getter of the log entry's tag.
- String Message `[get]`
  A getter of the log entry's message.
- UInt64 Timestamp `[get]`
  A getter of the log entry's timestamp.
- String ThreadID `[get]`
  A getter of the log entry's thread ID.

Additional Inherited Members

7.16.1 Detailed Description

A log entry data structure.

7.16.2 Member Function Documentation

7.16.2.1 Clone()

```
LogEntry MVCommon.LogEntry.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.16.3 Property Documentation

7.16.3.1 Timestamp

```
UInt64 MVCommon.LogEntry.Timestamp  [get]
```

A getter of the log entry's timestamp.

Timestamp is stored as number of milliseconds since the epoch - 1970-01-01 00:00:00.000.

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

- public/logger/LogEntry.cs

7.17 MVCommon.Logger Class Reference

A logger.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Logger >.

Public Member Functions

- Logger (LoggerLogLevel logLevel=LoggerLogLevel.LLL_DEBUG)
  A constructor.
- Logger (IntPtr nativeObject)
  A constructor.
- void AddLoggerSink (ILoggerSink loggerSink)
  Registers a logger sink.
- void RemoveLoggerSink (ILoggerSink loggerSink)
  Unregisters a logger sink.
- void RemoveAllLoggerSinks ()
  Unregisters all logger sinks.
- void LogMessage (LogLevel level, String tag, String format, params object[ ] args)
  Logs a new message.
- void LogMessage (UInt64 timestamp, String threadID, LogLevel level, String tag, String format, params object[ ] args)
  Logs a new message.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeLoggerObject `[get]`

  A getter of the native Logger object.
- LoggerLogLevel LogLevel `[get, set]`

  A property for setting and accessing log level of the logger for log messages filtering.

Additional Inherited Members

7.17.1 Detailed Description

A logger.

A logger receives requests for messages logging, filters them according to its log level setting and asynchronously pushes them to all attached logger sinks for customized handling.

7.17.2 Constructor & Destructor Documentation

7.17.2.1 Logger() [1/2]

```
MVCommon.Logger.Logger (
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_DEBUG )
```

A constructor.

Parameters

logLevel	an initial log level (default value -> debug and higher level messages are processed)

7.17.2.2 Logger() [2/2]

```
MVCommon.Logger.Logger (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native Logger object

7.17.3 Member Function Documentation

7.17.3.1 AddLoggerSink()

```
void MVCommon.Logger.AddLoggerSink (
            ILoggerSink loggerSink )
```

Registers a logger sink.

Parameters

loggerSink	a logger sink to register

7.17.3.2 LogMessage() [1/2]

```
void MVCommon.Logger.LogMessage (
            LogLevel level,
            String tag,
            String format,
            params object[] args )
```

Logs a new message.

Parameters

level	a level of the log message
tag	a tag of the log message
format	a formatting directive of the message in case there are additional arguments
args	arguments of the log message

A log level of the entry is compared with log level of the sink so entries with lesser log levels than the sink's log level are ignored.

Current time and the caller's thread is used for the log message.

7.17.3.3 LogMessage() [2/2]

```
void MVCommon.Logger.LogMessage (
            UInt64 timestamp,
            String threadID,
            LogLevel level,
            String tag,
            String format,
            params object[] args )
```

Logs a new message.

Parameters

timestamp	a timestamp of the log message (as number of milliseconds since the epoch - 1970-01-01 00:00:00.000)
threadID	a thread ID of the log message
level	a level of the log message
tag	a tag of the log message
format	a formatting directive of the message in case there are additional arguments
args	arguments of the log message

A log level of the entry is compared with log level of the sink so entries with lesser log levels than the sink's log level are ignored.

7.17.3.4 RemoveLoggerSink()

```
void MVCommon.Logger.RemoveLoggerSink (
            ILoggerSink loggerSink )
```

Unregisters a logger sink.

Parameters

loggerSink	a logger sink to unregister

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

- public/logger/Logger.cs

7.18 MVCommon.LoggerRegistry Class Reference

A global registry of loggers.

Static Public Member Functions

- static void RegisterLogger (String loggerAlias, Logger logger)

  Registers a logger instance to the registry.
- static void UnregisterLogger (String loggerAlias)

  Unregisters a logger registered with an alias from the registry.
- static Logger GetLogger (String loggerAlias)

  Returns a logger registered with an alias.
- static void ClearRegistry ()

  Clears the registry - removes all registered loggers.

7.18.1 Detailed Description

A global registry of loggers.

Serves as a global accessor to Logger instances in cases where direct access is not possible.

7.18.2 Member Function Documentation

7.18.2.1 GetLogger()

```
static Logger MVCommon.LoggerRegistry.GetLogger (
            String loggerAlias )  [static]
```

Returns a logger registered with an alias.

Parameters

loggerAlias	an alias the logger to return is supposed to be registered with

Returns

a logger with the alias or nullptr if there is none

7.18.2.2 RegisterLogger()

```
static void MVCommon.LoggerRegistry.RegisterLogger (
            String loggerAlias,
            Logger logger )  [static]
```

Registers a logger instance to the registry.

Parameters

loggerAlias	an alias to register the logger with
logger	a logger to register

Replaces the previous logger registered with the same alias in case there was one.

7.18.2.3 UnregisterLogger()

```
static void MVCommon.LoggerRegistry.UnregisterLogger (
            String loggerAlias )  [static]
```

Unregisters a logger registered with an alias from the registry.

Parameters

loggerAlias	an alias to unregister a logger registered with

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

- public/logger/LoggerRegistry.cs

7.19 MVCommon.Math Class Reference

A utility class for math operations.

Static Public Member Functions

- static bool AlmostEqual (float val1, float val2, float precision=0.001f)
  Compares two single-precision floating-point values with a tolerance.
- static bool AlmostEqual (double val1, double val2, double precision=0.0000001)
  Compares two double-precision floating-point values with a tolerance.

7.19.1 Detailed Description

A utility class for math operations.

7.19.2 Member Function Documentation

7.19.2.1 AlmostEqual() [1/2]

```
static bool MVCommon.Math.AlmostEqual (
            double val1,
            double val2,
            double precision = 0.0000001 )  [static]
```

Compares two double-precision floating-point values with a tolerance.

Parameters

val1	a value to compare
val2	a value to compare
precision	a required precision

Returns

true in case the difference between the two values is less or equal than a very small value (epsilon)

7.19.2.2 AlmostEqual() [2/2]

```
static bool MVCommon.Math.AlmostEqual (
            float val1,
            float val2,
            float precision = 0.001f )  [static]
```

Compares two single-precision floating-point values with a tolerance.

Parameters

val1	a value to compare
val2	a value to compare
precision	a required precision

Returns

true in case the difference between the two values is less or equal than a very small value (epsilon)

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

- public/math/Math.cs

7.20 MVCommon.Matrix4x4d Class Reference

A 4x4 matrix with double-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Matrix4x4d >.

Public Member Functions

- Matrix4x4d ()

  A constructor of an identity matrix (with all elements on main diagonal set to 1 and the rest set to 0).
- Matrix4x4d (double a00, double a01, double a02, double a03, double a10, double a11, double a12, double a13, double a20, double a21, double a22, double a23, double a30, double a31, double a32, double a33)

  A constructor.
- Matrix4x4d (Vector4d row0, Vector4d row1, Vector4d row2, Vector4d row3)

  A constructor.
- Matrix4x4d (IntPtr nativeObject)

  A constructor.
- String ToCommonString ()

  Converts the matrix into a human-readable string.
- void ToRawBytes (ByteArray bytes)

  Serializes the matrix into a byte array.
- double[ ] ToRawElements ()

  Serializes the matrix into an elements array.
- Matrix4x4d Transposed ()

  Creates a transposed matrix.
- Matrix4x4d Inverted ()

  Creates an inverted matrix.
- Matrix4x4d RotationTranslationMatrixInverted ()

  Creates an inverted matrix of a rotation-translation matrix.
- Matrix4x4d Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Matrix4x4d FromString (String str)

  Creates a matrix from a human-readable string.
- static Matrix4x4d FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes matrix from a byte array.
- static Matrix4x4d FromRawElements (double[ ] elements)

  Deserializes matrix from an elements array.
- static Matrix4x4d CreateZero ()

  Creates a matrix with all elements set to zero.
- static Matrix4x4d CreateTranslation (Vector3d translation)

  Creates a translation matrix.
- static Matrix4x4d CreateScale (Vector3d scale)

  Creates a scaling matrix.
- static Matrix4x4d CreateRotationFromEulerAnglesZYX (Vector3d eulerAngles)

  Creates a rotation matrix from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.
- static Matrix4x4d CreateRotationAroundAxis (Vector3d axis, double angle)

  Creates a rotation matrix from axis of rotation and an angle (in degrees).
- static Matrix4x4d CreateRotationFromVersor (Versord versor)

  Creates a rotation matrix from a versor.
- static Matrix4x4d CreateOrtographic (double left, double right, double bottom, double top, double near, double far)

  Creates a matrix for ortographic projection.
- static Matrix4x4d CreatePerspective (double fieldOfView, double aspectRatio, double near, double far)

  Creates a matrix for perspective projection.
- static Matrix4x4d CreateLookAt (Vector3d eyePosition, Vector3d centerPoint, Vector3d upDirection)

  Creates a viewing transformation matrix.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Matrix4x4d_GetRawBytesSize()

  A constant indicating the size of raw bytes of the matrix.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- double this[UInt64 row, UInt64 column] `[get, set]`

  Accesses a specific element in the matrix via indices.
- IntPtr nativeMatrixObject `[get]`

  A getter of the native matrix object.

Additional Inherited Members

7.20.1 Detailed Description

A 4x4 matrix with double-precision floating-point values.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 Matrix4x4d() [1/3]

```
MVCommon.Matrix4x4d.Matrix4x4d (
            double a00,
            double a01,
            double a02,
            double a03,
            double a10,
            double a11,
            double a12,
            double a13,
            double a20,
            double a21,
            double a22,
            double a23,
            double a30,
            double a31,
            double a32,
            double a33 )
```

A constructor.

Parameters

a00	an m[0][0] value
a01	an m[0][1] value
a02	an m[0][2] value
a03	an m[0][3] value
a10	an m[1][0] value
a11	an m[1][1] value
a12	an m[1][2] value
a13	an m[1][3] value
a20	an m[2][0] value
a21	an m[2][1] value
a22	an m[2][2] value
a23	an m[2][3] value
a30	an m[3][0] value
a31	an m[3][1] value
a32	an m[3][2] value
a33	an m[3][3] value

7.20.2.2 Matrix4x4d() [2/3]

```
MVCommon.Matrix4x4d.Matrix4x4d (
            Vector4d row0,
            Vector4d row1,
            Vector4d row2,
            Vector4d row3 )
```

A constructor.

Parameters

row0	a row 0
row1	a row 1
row2	a row 2
row3	a row 3

7.20.2.3 Matrix4x4d() [3/3]

```
MVCommon.Matrix4x4d.Matrix4x4d (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native matrix object

7.20.3 Member Function Documentation

7.20.3.1 Clone()

```
Matrix4x4d MVCommon.Matrix4x4d.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.20.3.2 CreateLookAt()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateLookAt (
            Vector3d eyePosition,
            Vector3d centerPoint,
            Vector3d upDirection )  [static]
```

Creates a viewing transformation matrix.

Parameters

eyePosition	a position of the viewing camera
centerPoint	a point the camera is looking at
upDirection	an up-direction of the viewing camera

Returns

a viewing transformation matrix

The resulting 'look-at' matrix follows the same principle as OpenGL's gluLookAt utility function: the matrix maps the reference (center) point to the negative z axis and the eye position to the origin. similarly, the up direction projected onto the viewing plane is mapped to the positive y axis. The UP vector must not be parallel to the line of sight from the eye position to the reference point.

7.20.3.3 CreateOrtographic()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateOrtographic (
            double left,
            double right,
            double bottom,
            double top,
            double near,
            double far )  [static]
```

Creates a matrix for ortographic projection.

Parameters

left	a left coordinate
right	a right coordinate
bottom	a bottom coordinate
top	a top coordinate
near	a near coordinate
far	a far coordinate

Returns

an ortographic-projection matrix

7.20.3.4 CreatePerspective()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreatePerspective (
            double fieldOfView,
            double aspectRatio,
            double near,
            double far )  [static]
```

Creates a matrix for perspective projection.

Parameters

fieldOfView	a field of view (in degrees)
aspectRatio	an aspect ratio
near	a near coordinate
far	a far coordinate

Returns

a perspective-projection matrix

7.20.3.5 CreateRotationAroundAxis()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateRotationAroundAxis (
            Vector3d axis,
            double angle )  [static]
```

Creates a rotation matrix from axis of rotation and an angle (in degrees).

Parameters

axis	an axis of rotation
angle	an angle

Returns

a rotation matrix

7.20.3.6 CreateRotationFromEulerAnglesZYX()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateRotationFromEulerAnglesZYX (
            Vector3d eulerAngles )  [static]
```

Creates a rotation matrix from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.

Parameters

eulerAngles	Euler angles of Z-Y-X rotation

Returns

a rotation matrix

7.20.3.7 CreateRotationFromVersor()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateRotationFromVersor (
            Versord versor )  [static]
```

Creates a rotation matrix from a versor.

Parameters

versor	a versor describing rotation

Returns

a rotation matrix

7.20.3.8 CreateScale()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateScale (
            Vector3d scale )  [static]
```

Creates a scaling matrix.

Parameters

scale	a vector describing the scale

Returns

a scaling matrix

7.20.3.9 CreateTranslation()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateTranslation (
            Vector3d translation )  [static]
```

Creates a translation matrix.

Parameters

translation	a vector describing the translation

Returns

a translation matrix

7.20.3.10 CreateZero()

```
static Matrix4x4d MVCommon.Matrix4x4d.CreateZero ( )  [static]
```

Creates a matrix with all elements set to zero.

Returns

a zero matrix

7.20.3.11 FromRawBytes()

```
static Matrix4x4d MVCommon.Matrix4x4d.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes matrix from a byte array.

Parameters

bytes	an array of matrix bytes
consumeBytes	determines whether bytes of the matrix shall be removed from the array

Returns

a matrix

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.20.3.12 FromRawElements()

```
static Matrix4x4d MVCommon.Matrix4x4d.FromRawElements (
            double[] elements )  [static]
```

Deserializes matrix from an elements array.

Parameters

elements	an array of 4x4 elements

Returns

a matrix

Exceptions

System.ArgumentException	raised when the elements array has less than 4x4 elements

7.20.3.13 FromString()

```
static Matrix4x4d MVCommon.Matrix4x4d.FromString (
            String str )  [static]
```

Creates a matrix from a human-readable string.

Parameters

str	a matrix string

Returns

a matrix

7.20.3.14 Inverted()

```
Matrix4x4d MVCommon.Matrix4x4d.Inverted ( )
```

Creates an inverted matrix.

Returns

an inverted matrix or null in case the creation failed, since it is not always possible to create one

7.20.3.15 RotationTranslationMatrixInverted()

```
Matrix4x4d MVCommon.Matrix4x4d.RotationTranslationMatrixInverted ( )
```

Creates an inverted matrix of a rotation-translation matrix.

It is always possible to create an inverted matrix of a rotation-translation matrix and the algorithm is much simpler and more effective than generic inversion algorithm. However, it is up to user to know what matrices he calls the function on

- the function assumes the matrix is a rotation-translation matrix.

  Returns

  an inverted matrix

7.20.3.16 ToCommonString()

```
String MVCommon.Matrix4x4d.ToCommonString ( )
```

Converts the matrix into a human-readable string.

Returns

the matrix string

7.20.3.17 ToRawBytes()

```
void MVCommon.Matrix4x4d.ToRawBytes (
            ByteArray bytes )
```

Serializes the matrix into a byte array.

Parameters

bytes	a byte array to serialize into

7.20.3.18 ToRawElements()

```
double [] MVCommon.Matrix4x4d.ToRawElements ( )
```

Serializes the matrix into an elements array.

Returns

an array of 4x4 elements

7.20.3.19 Transposed()

```
Matrix4x4d MVCommon.Matrix4x4d.Transposed ( )
```

Creates a transposed matrix.

Returns

a transposed matrix

7.20.4 Property Documentation

7.20.4.1 this[UInt64 row, UInt64 column]

```
double MVCommon.Matrix4x4d.this[UInt64 row, UInt64 column]  [get], [set]
```

Accesses a specific element in the matrix via indices.

Parameters

row	an index of the row to access
column	an index of the column to access

Returns

an element value

Exceptions

System.IndexOutOfRangeException	raised when indices are out of range (0-3)

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

- public/math/Matrix4x4d.cs

7.21 MVCommon.Matrix4x4f Class Reference

A 4x4 matrix with single-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Matrix4x4f >.

Public Member Functions

- Matrix4x4f ()

  A constructor of an identity matrix (with all elements on main diagonal set to 1 and the rest set to 0).
- Matrix4x4f (float a00, float a01, float a02, float a03, float a10, float a11, float a12, float a13, float a20, float a21, float a22, float a23, float a30, float a31, float a32, float a33)

  A constructor.
- Matrix4x4f (Vector4f row0, Vector4f row1, Vector4f row2, Vector4f row3)

  A constructor.
- Matrix4x4f (IntPtr nativeObject)

  A constructor.
- String ToCommonString ()

  Converts the matrix into a human-readable string.
- void ToRawBytes (ByteArray bytes)

  Serializes the matrix into a byte array.
- float[ ] ToRawElements ()

  Serializes the matrix into an elements array.
- Matrix4x4f Transposed ()

  Creates a transposed matrix.
- Matrix4x4f Inverted ()

  Creates an inverted matrix.
- Matrix4x4f RotationTranslationMatrixInverted ()

  Creates an inverted matrix of a rotation-translation matrix.
- Matrix4x4f Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Matrix4x4f FromString (String str)

  Creates a matrix from a human-readable string.
- static Matrix4x4f FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes matrix from a byte array.
- static Matrix4x4f FromRawElements (float[ ] elements)

  Deserializes matrix from an elements array.
- static Matrix4x4f CreateZero ()

  Creates a matrix with all elements set to zero.
- static Matrix4x4f CreateTranslation (Vector3f translation)

  Creates a translation matrix.
- static Matrix4x4f CreateScale (Vector3f scale)

  Creates a scaling matrix.
- static Matrix4x4f CreateRotationFromEulerAnglesZYX (Vector3f eulerAngles)

  Creates a rotation matrix from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.
- static Matrix4x4f CreateRotationAroundAxis (Vector3f axis, float angle)

  Creates a rotation matrix from axis of rotation and an angle (in degrees).
- static Matrix4x4f CreateRotationFromVersor (Versorf versor)

  Creates a rotation matrix from a versor.
- static Matrix4x4f CreateOrtographic (float left, float right, float bottom, float top, float near, float far)

  Creates a matrix for ortographic projection.
- static Matrix4x4f CreatePerspective (float fieldOfView, float aspectRatio, float near, float far)

  Creates a matrix for perspective projection.
- static Matrix4x4f CreateLookAt (Vector3f eyePosition, Vector3f centerPoint, Vector3f upDirection)

  Creates a viewing transformation matrix.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Matrix4x4f_GetRawBytesSize()

  A constant indicating the size of raw bytes of the matrix.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- float this[UInt64 row, UInt64 column] `[get, set]`

  Accesses a specific element in the matrix via indices.
- IntPtr nativeMatrixObject `[get]`

  A getter of the native matrix object.

Additional Inherited Members

7.21.1 Detailed Description

A 4x4 matrix with single-precision floating-point values.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 Matrix4x4f() [1/3]

```
MVCommon.Matrix4x4f.Matrix4x4f (
            float a00,
            float a01,
            float a02,
            float a03,
            float a10,
            float a11,
            float a12,
            float a13,
            float a20,
            float a21,
            float a22,
            float a23,
            float a30,
            float a31,
            float a32,
            float a33 )
```

A constructor.

Parameters

a00	an m[0][0] value
a01	an m[0][1] value
a02	an m[0][2] value
a03	an m[0][3] value
a10	an m[1][0] value
a11	an m[1][1] value
a12	an m[1][2] value
a13	an m[1][3] value
a20	an m[2][0] value
a21	an m[2][1] value
a22	an m[2][2] value
a23	an m[2][3] value
a30	an m[3][0] value
a31	an m[3][1] value
a32	an m[3][2] value
a33	an m[3][3] value

7.21.2.2 Matrix4x4f() [2/3]

```
MVCommon.Matrix4x4f.Matrix4x4f (
            Vector4f row0,
```

```
            Vector4f row1,
            Vector4f row2,
            Vector4f row3 )
```

A constructor.

Parameters

row0	a row 0
row1	a row 1
row2	a row 2
row3	a row 3

7.21.2.3 Matrix4x4f() [3/3]

```
MVCommon.Matrix4x4f.Matrix4x4f (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native matrix object

7.21.3 Member Function Documentation

7.21.3.1 Clone()

```
Matrix4x4f MVCommon.Matrix4x4f.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.21.3.2 CreateLookAt()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateLookAt (
            Vector3f eyePosition,
            Vector3f centerPoint,
            Vector3f upDirection )  [static]
```

Creates a viewing transformation matrix.

Parameters

eyePosition	a position of the viewing camera
centerPoint	a point the camera is looking at
upDirection	an up-direction of the viewing camera

Returns

a viewing transformation matrix

The resulting 'look-at' matrix follows the same principle as OpenGL's gluLookAt utility function: the matrix maps the reference (center) point to the negative z axis and the eye position to the origin. similarly, the up direction projected onto the viewing plane is mapped to the positive y axis. The UP vector must not be parallel to the line of sight from the eye position to the reference point.

7.21.3.3 CreateOrtographic()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateOrtographic (
            float left,
            float right,
            float bottom,
            float top,
            float near,
            float far )  [static]
```

Creates a matrix for ortographic projection.

Parameters

left	a left coordinate
right	a right coordinate
bottom	a bottom coordinate
top	a top coordinate
near	a near coordinate
far	a far coordinate

Returns

an ortographic-projection matrix

7.21.3.4 CreatePerspective()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreatePerspective (
            float fieldOfView,
            float aspectRatio,
            float near,
            float far )  [static]
```

Creates a matrix for perspective projection.

Parameters

fieldOfView	a field of view (in degrees)
aspectRatio	an aspect ratio
near	a near coordinate
far	a far coordinate

Returns

a perspective-projection matrix

7.21.3.5 CreateRotationAroundAxis()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateRotationAroundAxis (
            Vector3f axis,
            float angle )  [static]
```

Creates a rotation matrix from axis of rotation and an angle (in degrees).

Parameters

axis	an axis of rotation
angle	an angle

Returns

a rotation matrix

7.21.3.6 CreateRotationFromEulerAnglesZYX()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateRotationFromEulerAnglesZYX (
            Vector3f eulerAngles )  [static]
```

Creates a rotation matrix from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.

Parameters

eulerAngles	Euler angles of Z-Y-X rotation

Returns

a rotation matrix

7.21.3.7 CreateRotationFromVersor()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateRotationFromVersor (
            Versorf versor )  [static]
```

Creates a rotation matrix from a versor.

Parameters

versor	a versor describing rotation

Returns

a rotation matrix

7.21.3.8 CreateScale()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateScale (
            Vector3f scale )  [static]
```

Creates a scaling matrix.

Parameters

scale	a vector describing the scale

Returns

a scaling matrix

7.21.3.9 CreateTranslation()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateTranslation (
            Vector3f translation )  [static]
```

Creates a translation matrix.

Parameters

translation	a vector describing the translation

Returns

a translation matrix

7.21.3.10 CreateZero()

```
static Matrix4x4f MVCommon.Matrix4x4f.CreateZero ( )  [static]
```

Creates a matrix with all elements set to zero.

Returns

a zero matrix

7.21.3.11 FromRawBytes()

```
static Matrix4x4f MVCommon.Matrix4x4f.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes matrix from a byte array.

Parameters

bytes	an array of matrix bytes
consumeBytes	determines whether bytes of the matrix shall be removed from the array

Returns

a matrix

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.21.3.12 FromRawElements()

```
static Matrix4x4f MVCommon.Matrix4x4f.FromRawElements (
            float[] elements )  [static]
```

Deserializes matrix from an elements array.

Parameters

elements	an array of 4x4 elements

Returns

a matrix

Exceptions

System.ArgumentException	raised when the elements array has less than 4x4 elements

7.21.3.13 FromString()

```
static Matrix4x4f MVCommon.Matrix4x4f.FromString (
            String str )  [static]
```

Creates a matrix from a human-readable string.

Parameters

str	a matrix string

Returns

a matrix

7.21.3.14 Inverted()

```
Matrix4x4f MVCommon.Matrix4x4f.Inverted ( )
```

Creates an inverted matrix.

Returns

an inverted matrix or null in case the creation failed, since it is not always possible to create one

7.21.3.15 RotationTranslationMatrixInverted()

```
Matrix4x4f MVCommon.Matrix4x4f.RotationTranslationMatrixInverted ( )
```

Creates an inverted matrix of a rotation-translation matrix.

It is always possible to create an inverted matrix of a rotation-translation matrix and the algorithm is much simpler and more effective than generic inversion algorithm. However, it is up to user to know what matrices he calls the function on

- the function assumes the matrix is a rotation-translation matrix.

  Returns

  an inverted matrix

7.21.3.16 ToCommonString()

```
String MVCommon.Matrix4x4f.ToCommonString ( )
```

Converts the matrix into a human-readable string.

Returns

the matrix string

7.21.3.17 ToRawBytes()

```
void MVCommon.Matrix4x4f.ToRawBytes (
            ByteArray bytes )
```

Serializes the matrix into a byte array.

Parameters

bytes	a byte array to serialize into

7.21.3.18 ToRawElements()

```
float [] MVCommon.Matrix4x4f.ToRawElements ( )
```

Serializes the matrix into an elements array.

Returns

an array of 4x4 elements

7.21.3.19 Transposed()

```
Matrix4x4f MVCommon.Matrix4x4f.Transposed ( )
```

Creates a transposed matrix.

Returns

a transposed matrix

7.21.4 Property Documentation

7.21.4.1 this[UInt64 row, UInt64 column]

```
float MVCommon.Matrix4x4f.this[UInt64 row, UInt64 column]  [get], [set]
```

Accesses a specific element in the matrix via indices.

Parameters

row	an index of the row to access
column	an index of the column to access

Returns

an element value

Exceptions

System.IndexOutOfRangeException	raised when indices are out of range (0-3)

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

- public/math/Matrix4x4f.cs

7.22 MonoPInvokeCallbackAttribute Class Reference

A redefinition of MonoPInvokeCallback attribute necessary for setting managed delegates as function pointers to native code on iOS platform.

Inherits Attribute.

Public Member Functions

- MonoPInvokeCallbackAttribute (Type t)

  A constructor.

7.22.1 Detailed Description

A redefinition of MonoPInvokeCallback attribute necessary for setting managed delegates as function pointers to native code on iOS platform.

7.22.2 Constructor & Destructor Documentation

7.22.2.1 MonoPInvokeCallbackAttribute()

```
MonoPInvokeCallbackAttribute.MonoPInvokeCallbackAttribute (
            Type t )
```

A constructor.

Parameters

t	a type of delegate

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

- public/MonoPInvokeCallback.cs

7.23 MVCommon.NativeObjectHolder Class Reference

A holder of a native object with support for proper object destruction.

Inherits IDisposable.

Inherited by MVCommon.BlockingCounter, MVCommon.ByteArray, MVCommon.CameraParams, MVCommon.Color, MVCommon.Guid, MVCommon.GuidAliasDatabase, MVCommon.IBlockingCounterCondition, MVCommon.ILoggerSink, MVCommon.IThreadPoolJob, MVCommon.LogEntry, MVCommon.Logger, MVCommon.Matrix4x4d, MVCommon.Matrix4x4f, MVCommon.String, MVCommon.ThreadPool, MVCommon.Vector2d, MVCommon.Vector2f, MVCommon.Vector3d, MVCommon.Vector3f, MVCommon.Vector4d, MVCommon.Vector4f, MVCommon.VersionInfo, MVCommon.Versord, and MVCommon.Versorf.

Public Member Functions

- NativeObjectHolder (IntPtr nativeObject)

  A constructor.

Protected Member Functions

- void Dispose (bool calledFromDispose)

  Disposes the unmanaged native object.
- abstract void DestroyNativeObject ()

  Destroys the native object in a customized way.

Protected Attributes

- object m_nativeObjectLock = new object()

  An object for protecting access to the native object pointer.

Properties

- IntPtr nativeObject `[get]`

  A getter of the native object.

7.23.1 Detailed Description

A holder of a native object with support for proper object destruction.

IDisposable implementation allows immediate release of the held native object (i.e. no waiting for garbage collection).

7.23.2 Constructor & Destructor Documentation

7.23.2.1 NativeObjectHolder()

```
MVCommon.NativeObjectHolder.NativeObjectHolder (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native object to hold

7.23.3 Member Function Documentation

7.23.3.1 Dispose()

```
void MVCommon.NativeObjectHolder.Dispose (
            bool calledFromDispose )  [protected]
```

Disposes the unmanaged native object.

Parameters

calledFromDispose	an indication of whether the caller is IDisposable.Dispose() or destructor - in case of the former, the destructor execution is suppressed for better performance, since the disposal of the native object was already done

7.23.4 Member Data Documentation

7.23.4.1 m_nativeObjectLock

```
object MVCommon.NativeObjectHolder.m_nativeObjectLock = new object()  [protected]
```

An object for protecting access to the native object pointer.

Derivatives may use the lock to ensure they do not perform any operation on for instance a native object that is just being destroyed.

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

- public/NativeObjectHolder.cs

7.24 MVCommon.NetBlockingCounterCondition Class Reference

An abstract blocking counter condition base class intended for creation of new condition implementations in .Net environment.

Inherits MVCommon.IBlockingCounterCondition.

Public Member Functions

- NetBlockingCounterCondition ()
  A constructor.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.
- abstract bool CheckCondition (Int32 value)
  A callback executed for checking the condition with a value.

Additional Inherited Members

7.24.1 Detailed Description

An abstract blocking counter condition base class intended for creation of new condition implementations in .Net environment.

7.24.2 Member Function Documentation

7.24.2.1 CheckCondition()

```
abstract bool MVCommon.NetBlockingCounterCondition.CheckCondition (
            Int32 value )  [protected], [pure virtual]
```

A callback executed for checking the condition with a value.

Parameters

value	a value the condition is checked with

Returns

true in case the condition passes with the value, false otherwise

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

- public/utils/blockingcounter/NetBlockingCounterCondition.cs

7.25 MVCommon.NetLoggerSink Class Reference

An abstract logger sink base class intended for creation of new logger sink implementations in .Net environment.

Inherits MVCommon.ILoggerSink.

Public Member Functions

- NetLoggerSink (LoggerLogLevel logLevel=LoggerLogLevel.LLL_VERBOSE)
  A constructor.

Static Public Member Functions

- static MVCommon.String TimestampToString (UInt64 timestamp, bool includeDate=true)
  A default formatter of timestamps usable in logger sink implementations.
- static MVCommon.String LogLevelToString (LogLevel level, bool shortVersion=false)
  A default formatter of log levels usable in logger sink implementations.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.
- abstract void HandleLogEntry (LogEntry logEntry)
  A callback executed when a new log entry is added.

Additional Inherited Members

7.25.1 Detailed Description

An abstract logger sink base class intended for creation of new logger sink implementations in .Net environment.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 NetLoggerSink()

```
MVCommon.NetLoggerSink.NetLoggerSink (
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_VERBOSE )
```

A constructor.

Parameters

logLevel	an initial log level (default value -> all log messages are processed)

7.25.3 Member Function Documentation

7.25.3.1 HandleLogEntry()

```
abstract void MVCommon.NetLoggerSink.HandleLogEntry (
            LogEntry logEntry )  [protected], [pure virtual]
```

A callback executed when a new log entry is added.

Parameters

logEntry	a new log entry

7.25.3.2 LogLevelToString()

```
static MVCommon.String MVCommon.NetLoggerSink.LogLevelToString (
            LogLevel level,
            bool shortVersion = false )  [static]
```

A default formatter of log levels usable in logger sink implementations.

Parameters

level	a log level to format
shortVersion	indicates whether a short (single character) or a long (whole level name) version shall be used

Returns

a formatted log level

7.25.3.3 TimestampToString()

```
static MVCommon.String MVCommon.NetLoggerSink.TimestampToString (
            UInt64 timestamp,
            bool includeDate = true )  [static]
```

A default formatter of timestamps usable in logger sink implementations.

A format of the resulting string is "1900-Jan-01 00:00:00.000" when date is included and "00:00:00.000" when date is ommited.

Parameters

timestamp	a timestamp to format
includeDate	indicates whether a date shall be included in the formatted string

Returns

a formatted timestamp

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

- public/logger/NetLoggerSink.cs

7.26 MVCommon.NetThreadPoolJob Class Reference

An abstract thread pool job base class intended for creation of new job implementations in .Net environment.

Inherits MVCommon.IThreadPoolJob.

Public Member Functions

- NetThreadPoolJob ()
  A constructor.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.
- abstract void Execute (UInt32 threadID)
  The job-executing operator.

Additional Inherited Members

7.26.1 Detailed Description

An abstract thread pool job base class intended for creation of new job implementations in .Net environment.

7.26.2 Member Function Documentation

7.26.2.1 Execute()

```
abstract void MVCommon.NetThreadPoolJob.Execute (
            UInt32 threadID )  [protected], [pure virtual]
```

The job-executing operator.

Parameters

threadID	an ID of the thread that executes the job

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

- public/utils/threadpool/NetThreadPoolJob.cs

7.27 MVCommon.RedirectingLoggerSink Class Reference

A logger sink implementation for redirecting log messages to another logger.

Inherits MVCommon.ILoggerSink.

Public Member Functions

- RedirectingLoggerSink (Logger logger, LoggerLogLevel logLevel=LoggerLogLevel.LLL_VERBOSE)
  A constructor.

Additional Inherited Members

7.27.1 Detailed Description

A logger sink implementation for redirecting log messages to another logger.

All properties of log messages (timestamp, thread ID, ...) received from a logger are preserved when redirected to the target logger.

7.27.2 Constructor & Destructor Documentation

7.27.2.1 RedirectingLoggerSink()

```
MVCommon.RedirectingLoggerSink.RedirectingLoggerSink (
            Logger logger,
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_VERBOSE )
```

A constructor.

Parameters

logger	a logger to redirect log messages to
logLevel	an initial log level (default value -> all log messages are processed)

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

- public/logger/sinks/RedirectingLoggerSink.cs

7.28 MVCommon.SharedRef< T > Class Template Reference

A smart reference implementation with object reference counting.

Inherits IDisposable.

Public Member Functions

- SharedRef (T sharedObj)
  A constructor.
- SharedRef (SharedRef< T > other)
  A copy constructor.
- void Dispose ()
  Disposes the shared reference.
- SharedRef< T > CloneRef ()
  Clones the shared reference.

Static Public Member Functions

- static SharedRef< T > Create (T sharedObj)
  A creator of the new shared reference.

Protected Member Functions

- virtual void Dispose (bool calledFromDispose)
  Disposes the shared reference.

Properties

- T? sharedObj `[get]`
  A getter of the shared object.

7.28.1 Detailed Description

A smart reference implementation with object reference counting.

As soon as the number of references to the shared object drops to 0, the object is disposed.

Template Parameters

T	a type of the shared object

Type Constraints

T* : *class

T* : *IDisposable

7.28.2 Constructor & Destructor Documentation

7.28.2.1 SharedRef() [1/2]

```
MVCommon.SharedRef< T >.SharedRef (
            T sharedObj )
```

A constructor.

Creates a new shared reference with count of references equal to 1.

Parameters

sharedObj	an object to become shared

7.28.2.2 SharedRef() [2/2]

```
MVCommon.SharedRef< T >.SharedRef (
            SharedRef< T > other )
```

A copy constructor.

Stores shared object of the other reference and increases references count by 1.

Parameters

other	another reference to share the object with

7.28.3 Member Function Documentation

7.28.3.1 CloneRef()

```
SharedRef<T> MVCommon.SharedRef< T >.CloneRef ( )
```

Clones the shared reference.

Increases the count of references to the shared object by 1.

Returns

a new shared reference to the same shared object

7.28.3.2 Create()

```
static SharedRef<T> MVCommon.SharedRef< T >.Create (
            T sharedObj )  [static]
```

A creator of the new shared reference.

Parameters

sharedObj	an object to become shared

Returns

a new shared reference, with references count equal to 1

7.28.3.3 Dispose() [1/2]

```
void MVCommon.SharedRef< T >.Dispose ( )
```

Disposes the shared reference.

Decreases the references count to the shared object by 1.

7.28.3.4 Dispose() [2/2]

```
virtual void MVCommon.SharedRef< T >.Dispose (
            bool calledFromDispose )  [protected], [virtual]
```

Disposes the shared reference.

Decreases the references count to the shared object by 1 and disposes the object if the count drops to 0.

Parameters

calledFromDispose	an indication of whether the call is made from destructor of from IDisposable.Dispose()

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

- public/SharedRef.cs

7.29 MVCommon.StdOutLoggerSink Class Reference

A logger sink implementation for logging into a standard output.

Inherits MVCommon.ILoggerSink.

Public Member Functions

- StdOutLoggerSink (bool colorizeByLevel=false, LoggerLogLevel logLevel=LoggerLogLevel.LLL_VERBOSE)
  A constructor.

Additional Inherited Members

7.29.1 Detailed Description

A logger sink implementation for logging into a standard output.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 StdOutLoggerSink()

```
MVCommon.StdOutLoggerSink.StdOutLoggerSink (
            bool colorizeByLevel = false,
            LoggerLogLevel logLevel = LoggerLogLevel.LLL_VERBOSE )
```

A constructor.

Parameters

colorizeByLevel	determines whether log messages shall be colorized based on their level - actual behaviour depends on the console used
logLevel	an initial log level (default value -> all log messages are processed)

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

- public/logger/sinks/StdOutLoggerSink.cs

7.30 MVCommon.String Class Reference

A string implementation.

Inherits MVCommon.NativeObjectHolder, and IEquatable< String >.

Public Member Functions

- String (string str="")

  A constructor.
- String (IntPtr nativeObject)

  A constructor.
- String Substr (Int32 pos=0, Int32 len=-1)

  Generates a substring of the string.
- String Clone ()

  Makes an independent clone object.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeStringObject `[get]`

  A getter of the native String object.
- string NetString `[get]`

  Constructs a .Net string from the MVCommon String.
- Int32 Length `[get]`

  Gets length of the string.
- char this[int i] `[get, set]`

  A property for accessing specific string element (character).

Additional Inherited Members

7.30.1 Detailed Description

A string implementation.

7.30.2 Constructor & Destructor Documentation

7.30.2.1 String() [1/2]

```
MVCommon.String.String (
            string str = "" )
```

A constructor.

Parameters

str	a string

7.30.2.2 String() [2/2]

```
MVCommon.String.String (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native String object

7.30.3 Member Function Documentation

7.30.3.1 Clone()

```
String MVCommon.String.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.30.3.2 Substr()

```
String MVCommon.String.Substr (
            Int32 pos = 0,
            Int32 len = -1 )
```

Generates a substring of the string.

Parameters

pos	a starting position of the string to generate the substring from
len	a length of the substring (special value -1 means the rest of the original string)

Returns

the string's substring

7.30.4 Property Documentation

7.30.4.1 Length

```
Int32 MVCommon.String.Length  [get]
```

Gets length of the string.

Returns

string's length

7.30.4.2 NetString

```
string MVCommon.String.NetString  [get]
```

Constructs a .Net string from the MVCommon String.

The returned .Net string is independent from the MVCommon string, so no modifications are reflected back.

Returns

a .Net string

7.30.4.3 this[int i]

```
char MVCommon.String.this[int i]  [get], [set]
```

A property for accessing specific string element (character).

Parameters

i	character's index

Returns

a character at the index

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

- public/utils/String.cs

7.31 MVCommon.ThreadPool Class Reference

A pool of threads.

Inherits MVCommon.NativeObjectHolder.

Public Member Functions

- ThreadPool (UInt32 threadsCount=1, Int32 waitersForUnoccupiedThreadsCountHint=1)
  A constructor.
- UInt32 GetThreadsCount ()
  Gets the threads count.
- bool DoJob (IThreadPoolJob job)
  Instructs the pool to execute a job on an unoccupied thread.
- bool HasUnoccupiedThreads ()
  Determines whether there are unoccupied threads available in the pool.
- UInt32 GetUnoccupiedThreadsCount ()
  Determines a count of unoccupied threads in the pool.
- void WaitForAnUnoccupiedThread ()
  Blocks current thread until there is at least one unoccupied thread in the pool.
- void ResetJobs ()
  Resets all jobs and threads executed by the pool.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.

Additional Inherited Members

7.31.1 Detailed Description

A pool of threads.

The pool maintains a fixed-size collection of threads usable for executing jobs. It hides the details about creation and maintenance of threads and about dispatching of jobs to them, allowing a user to focus on the jobs themselves.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 ThreadPool()

```
MVCommon.ThreadPool.ThreadPool (
            UInt32 threadsCount = 1,
            Int32 waitersForUnoccupiedThreadsCountHint = 1 )
```

A constructor.

Instantiates the threads.

Parameters

threadsCount	a count of threads the pool maintains
waitersForUnoccupiedThreadsCountHint	a hint about expected count of threads calling WaitForAnUnoccupiedThread - it allows an optimization of internal memory allocations made per each waiting call in cases when count of parallel waiters can be predicted. Special value `0` will result in allocations made every time, and `negative` hint value results in no deallocations (and thus maximum reusability of the memory) during the entire lifetime of the thread pool.

7.31.3 Member Function Documentation

7.31.3.1 DestroyNativeObject()

```
override void MVCommon.ThreadPool.DestroyNativeObject ( )  [protected], [virtual]
```

Destroys the native object in a customized way.

Shuts down all maintained threads and waits until they complete their execution.

Implements MVCommon.NativeObjectHolder.

7.31.3.2 DoJob()

```
bool MVCommon.ThreadPool.DoJob (
            IThreadPoolJob job )
```

Instructs the pool to execute a job on an unoccupied thread.

Parameters

job	a job to execute

Returns

true in case there is an unoccupied thread that will execute the job, false otherwise

7.31.3.3 GetThreadsCount()

```
UInt32 MVCommon.ThreadPool.GetThreadsCount ( )
```

Gets the threads count.

Returns

the count of threads in the pool

7.31.3.4 GetUnoccupiedThreadsCount()

```
UInt32 MVCommon.ThreadPool.GetUnoccupiedThreadsCount ( )
```

Determines a count of unoccupied threads in the pool.

Returns

a count of currently unoccupied threads

7.31.3.5 HasUnoccupiedThreads()

```
bool MVCommon.ThreadPool.HasUnoccupiedThreads ( )
```

Determines whether there are unoccupied threads available in the pool.

Returns

true if there is at least one unoccupied thread, false otherwise

7.31.3.6 ResetJobs()

```
void MVCommon.ThreadPool.ResetJobs ( )
```

Resets all jobs and threads executed by the pool.

Waits until all jobs are completed, shuts down all the threads and reinitializes them.

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

- public/utils/threadpool/ThreadPool.cs

7.32 MVCommon.Vector2d Class Reference

A 2-dimensional vector with double-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Vector2d >.

Public Member Functions

- Vector2d ()

  A default constructor.
- Vector2d (double x, double y)

  A constructor.
- Vector2d (IntPtr nativeObject)

  A constructor.
- String ToCommonString ()

  Converts the vector into a human-readable string.
- void ToRawBytes (ByteArray bytes)

  Serializes the vector into a byte array.
- double Length ()

  Gets a length of the vector.
- Vector2d Inverted ()

  Creates a vector with inverted dimensions (1/x).
- Vector2d Normalized ()

  Creates a normalized vector (with length equal to 1).
- Vector2d Abs ()

  Creates a vector with dimensions with absolute values of the original vector.
- Vector2d Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Vector2d FromString (String str)

  Creates a vector from a human-readable string.
- static Vector2d FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes vector from a byte array.
- static double Dot (Vector2d lhs, Vector2d rhs)

  Calculates a dot product of two vectors.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Vector2d_GetRawBytesSize()

  A constant indicating the size of raw bytes of the vector.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- double x `[get, set]`

  An x coordinate.
- double y `[get, set]`

  A y coordinate.
- double this[int i] `[get, set]`

  Accesses vector dimension value via index.
- IntPtr nativeVectorObject `[get]`

  A getter of the native vector object.

Additional Inherited Members

7.32.1 Detailed Description

A 2-dimensional vector with double-precision floating-point values.

7.32.2 Constructor & Destructor Documentation

7.32.2.1 Vector2d() [1/2]

```
MVCommon.Vector2d.Vector2d (
            double x,
            double y )
```

A constructor.

Parameters

x	an x coordinate
y	a y coordinate

7.32.2.2 Vector2d() [2/2]

```
MVCommon.Vector2d.Vector2d (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native vector object

7.32.3 Member Function Documentation

7.32.3.1 Abs()

```
Vector2d MVCommon.Vector2d.Abs ( )
```

Creates a vector with dimensions with absolute values of the original vector.

Returns

a vector with absolute-valued dimensions

7.32.3.2 Clone()

```
Vector2d MVCommon.Vector2d.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.32.3.3 Dot()

```
static double MVCommon.Vector2d.Dot (
            Vector2d lhs,
            Vector2d rhs )  [static]
```

Calculates a dot product of two vectors.

Parameters

lhs	a first vector-operand
rhs	a second vector-operand

Returns

a dot product

7.32.3.4 FromRawBytes()

```
static Vector2d MVCommon.Vector2d.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes vector from a byte array.

Parameters

bytes	an array of vector bytes
consumeBytes	determines whether bytes of the vector shall be removed from the array

Returns

a vector

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.32.3.5 FromString()

```
static Vector2d MVCommon.Vector2d.FromString (
            String str )  [static]
```

Creates a vector from a human-readable string.

Parameters

str	a vector string

Returns

a vector

7.32.3.6 Inverted()

```
Vector2d MVCommon.Vector2d.Inverted ( )
```

Creates a vector with inverted dimensions (1/x).

Returns

an inverted vector

7.32.3.7 Length()

```
double MVCommon.Vector2d.Length ( )
```

Gets a length of the vector.

Returns

vector's length

7.32.3.8 Normalized()

```
Vector2d MVCommon.Vector2d.Normalized ( )
```

Creates a normalized vector (with length equal to 1).

Returns an unchanged vector in case its length is equal to 0.

Returns

a normalized vector

7.32.3.9 ToCommonString()

```
String MVCommon.Vector2d.ToCommonString ( )
```

Converts the vector into a human-readable string.

Returns

the vector string

7.32.3.10 ToRawBytes()

```
void MVCommon.Vector2d.ToRawBytes (
            ByteArray bytes )
```

Serializes the vector into a byte array.

Parameters

bytes	a byte array to serialize into

7.32.4 Property Documentation

7.32.4.1 this[int i]

```
double MVCommon.Vector2d.this[int i]  [get], [set]
```

Accesses vector dimension value via index.

Parameters

i	an index of the dimension to access

Returns

a dimension value

Exceptions

System.IndexOutOfRangeException	raised when index is out of range (0-1)

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

- public/math/Vector2d.cs

7.33 MVCommon.Vector2f Class Reference

A 2-dimensional vector with single-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Vector2f >.

Public Member Functions

- Vector2f ()
  A default constructor.
- Vector2f (float x, float y)
  A constructor.
- Vector2f (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the vector into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the vector into a byte array.
- float Length ()
  Gets a length of the vector.

- Vector2f Inverted ()

  Creates a vector with inverted dimensions (1/x).
- Vector2f Normalized ()

  Creates a normalized vector (with length equal to 1).
- Vector2f Abs ()

  Creates a vector with dimensions with absolute values of the original vector.
- Vector2f Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Vector2f FromString (String str)

  Creates a vector from a human-readable string.
- static Vector2f FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes vector from a byte array.
- static float Dot (Vector2f lhs, Vector2f rhs)

  Calculates a dot product of two vectors.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Vector2f_GetRawBytesSize()

  A constant indicating the size of raw bytes of the vector.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- float x `[get, set]`

  An x coordinate.
- float y `[get, set]`

  A y coordinate.
- float this[int i] `[get, set]`

  Accesses vector dimension value via index.
- IntPtr nativeVectorObject `[get]`

  A getter of the native vector object.

Additional Inherited Members

7.33.1 Detailed Description

A 2-dimensional vector with single-precision floating-point values.

7.33.2 Constructor & Destructor Documentation

7.33.2.1 Vector2f() [1/2]

```
MVCommon.Vector2f.Vector2f (
            float x,
            float y )
```

A constructor.

Parameters

x	an x coordinate
y	a y coordinate

7.33.2.2 Vector2f() [2/2]

```
MVCommon.Vector2f.Vector2f (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native vector object

7.33.3 Member Function Documentation

7.33.3.1 Abs()

```
Vector2f MVCommon.Vector2f.Abs ( )
```

Creates a vector with dimensions with absolute values of the original vector.

Returns

a vector with absolute-valued dimensions

7.33.3.2 Clone()

```
Vector2f MVCommon.Vector2f.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.33.3.3 Dot()

```
static float MVCommon.Vector2f.Dot (
            Vector2f lhs,
            Vector2f rhs )  [static]
```

Calculates a dot product of two vectors.

Parameters

lhs	a first vector-operand
rhs	a second vector-operand

Returns

a dot product

7.33.3.4 FromRawBytes()

```
static Vector2f MVCommon.Vector2f.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes vector from a byte array.

Parameters

bytes	an array of vector bytes
consumeBytes	determines whether bytes of the vector shall be removed from the array

Returns

a vector

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.33.3.5 FromString()

```
static Vector2f MVCommon.Vector2f.FromString (
            String str )  [static]
```

Creates a vector from a human-readable string.

Parameters

str	a vector string

Returns

a vector

7.33.3.6 Inverted()

```
Vector2f MVCommon.Vector2f.Inverted ( )
```

Creates a vector with inverted dimensions (1/x).

Returns

an inverted vector

7.33.3.7 Length()

```
float MVCommon.Vector2f.Length ( )
```

Gets a length of the vector.

Returns

vector's length

7.33.3.8 Normalized()

```
Vector2f MVCommon.Vector2f.Normalized ( )
```

Creates a normalized vector (with length equal to 1).

Returns an unchanged vector in case its length is equal to 0.

Returns

a normalized vector

7.33.3.9 ToCommonString()

```
String MVCommon.Vector2f.ToCommonString ( )
```

Converts the vector into a human-readable string.

Returns

the vector string

7.33.3.10 ToRawBytes()

```
void MVCommon.Vector2f.ToRawBytes (
            ByteArray bytes )
```

Serializes the vector into a byte array.

Parameters

bytes	a byte array to serialize into

7.33.4 Property Documentation

7.33.4.1 this[int i]

```
float MVCommon.Vector2f.this[int i]  [get], [set]
```

Accesses vector dimension value via index.

Parameters

i	an index of the dimension to access

Returns

a dimension value

Exceptions

System.IndexOutOfRangeException	raised when index is out of range (0-1)

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

- public/math/Vector2f.cs

7.34 MVCommon.Vector3d Class Reference

A 3-dimensional vector with double-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Vector3d >.

Public Member Functions

- Vector3d ()
  A default constructor.
- Vector3d (double x, double y, double z)
  A constructor.
- Vector3d (Vector2d vector2, double z=0.0)
  A constructor.
- Vector3d (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the vector into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the vector into a byte array.
- double Length ()
  Gets a length of the vector.
- Vector3d Inverted ()
  Creates a vector with inverted dimensions (1/x).
- Vector3d Normalized ()
  Creates a normalized vector (with length equal to 1).
- Vector3d Abs ()
  Creates a vector with dimensions with absolute values of the original vector.
- Vector2d GetXY ()
  Extracts x and y coordinates as a 2-dimensional vector.
- Vector3d Clone ()
  Makes an independent clone object.

Static Public Member Functions

- static Vector3d FromString (String str)

  Creates a vector from a human-readable string.
- static Vector3d FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes vector from a byte array.
- static double Dot (Vector3d lhs, Vector3d rhs)

  Calculates a dot product of two vectors.
- static Vector3d Cross (Vector3d lhs, Vector3d rhs)

  Calculates a cross product of two vectors.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Vector3d_GetRawBytesSize()

  A constant indicating the size of raw bytes of the vector.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- double x `[get, set]`

  An x coordinate.
- double y `[get, set]`

  A y coordinate.
- double z `[get, set]`

  A z coordinate.
- double this[int i] `[get, set]`

  Accesses vector dimension value via index.
- IntPtr nativeVectorObject `[get]`

  A getter of the native vector object.

Additional Inherited Members

7.34.1 Detailed Description

A 3-dimensional vector with double-precision floating-point values.

7.34.2 Constructor & Destructor Documentation

7.34.2.1 Vector3d() [1/3]

```
MVCommon.Vector3d.Vector3d (
            double x,
            double y,
            double z )
```

A constructor.

Parameters

x	an x coordinate
y	a y coordinate
z	a z coordinate

7.34.2.2 Vector3d() [2/3]

```
MVCommon.Vector3d.Vector3d (
            Vector2d vector2,
            double z = 0.0 )
```

A constructor.

Parameters

vector2	a 2-dimensional vector whose x and y coordinates will be grabbed
z	a z coordinate

7.34.2.3 Vector3d() [3/3]

```
MVCommon.Vector3d.Vector3d (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native vector object

7.34.3 Member Function Documentation

7.34.3.1 Abs()

```
Vector3d MVCommon.Vector3d.Abs ( )
```

Creates a vector with dimensions with absolute values of the original vector.

Returns

a vector with absolute-valued dimensions

7.34.3.2 Clone()

```
Vector3d MVCommon.Vector3d.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.34.3.3 Cross()

```
static Vector3d MVCommon.Vector3d.Cross (
            Vector3d lhs,
            Vector3d rhs )  [static]
```

Calculates a cross product of two vectors.

Parameters

lhs	a left-hand-side vector-operand
rhs	a right-hand-side vector-operand

Returns

a vector representing the cross product

7.34.3.4 Dot()

```
static double MVCommon.Vector3d.Dot (
            Vector3d lhs,
            Vector3d rhs )  [static]
```

Calculates a dot product of two vectors.

Parameters

lhs	a first vector-operand
rhs	a second vector-operand

Returns

a dot product

7.34.3.5 FromRawBytes()

```
static Vector3d MVCommon.Vector3d.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes vector from a byte array.

Parameters

bytes	an array of vector bytes
consumeBytes	determines whether bytes of the vector shall be removed from the array

Returns

a vector

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.34.3.6 FromString()

```
static Vector3d MVCommon.Vector3d.FromString (
            String str )  [static]
```

Creates a vector from a human-readable string.

Parameters

str	a vector string

Returns

a vector

7.34.3.7 GetXY()

```
Vector2d MVCommon.Vector3d.GetXY ( )
```

Extracts x and y coordinates as a 2-dimensional vector.

Returns

a 2-dimensional vector

7.34.3.8 Inverted()

```
Vector3d MVCommon.Vector3d.Inverted ( )
```

Creates a vector with inverted dimensions (1/x).

Returns

an inverted vector

7.34.3.9 Length()

```
double MVCommon.Vector3d.Length ( )
```

Gets a length of the vector.

Returns

vector's length

7.34.3.10 Normalized()

```
Vector3d MVCommon.Vector3d.Normalized ( )
```

Creates a normalized vector (with length equal to 1).

Returns an unchanged vector in case its length is equal to 0.

Returns

a normalized vector

7.34.3.11 ToCommonString()

```
String MVCommon.Vector3d.ToCommonString ( )
```

Converts the vector into a human-readable string.

Returns

the vector string

7.34.3.12 ToRawBytes()

```
void MVCommon.Vector3d.ToRawBytes (
            ByteArray bytes )
```

Serializes the vector into a byte array.

Parameters

bytes	a byte array to serialize into

7.34.4 Property Documentation

7.34.4.1 this[int i]

```
double MVCommon.Vector3d.this[int i]  [get], [set]
```

Accesses vector dimension value via index.

Parameters

i	an index of the dimension to access

Returns

a dimension value

Exceptions

System.IndexOutOfRangeException	raised when index is out of range (0-2)

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

- public/math/Vector3d.cs

7.35 MVCommon.Vector3f Class Reference

A 3-dimensional vector with single-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Vector3f >.

Public Member Functions

- Vector3f ()

  A default constructor.
- Vector3f (float x, float y, float z)

  A constructor.
- Vector3f (Vector2f vector2, float z=0.0f)

  A constructor.
- Vector3f (IntPtr nativeObject)

  A constructor.
- String ToCommonString ()

  Converts the vector into a human-readable string.
- void ToRawBytes (ByteArray bytes)

  Serializes the vector into a byte array.
- float Length ()

  Gets a length of the vector.
- Vector3f Inverted ()

  Creates a vector with inverted dimensions (1/x).
- Vector3f Normalized ()

  Creates a normalized vector (with length equal to 1).
- Vector3f Abs ()

  Creates a vector with dimensions with absolute values of the original vector.
- Vector2f GetXY ()

  Extracts x and y coordinates as a 2-dimensional vector.
- Vector3f Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Vector3f FromString (String str)

  Creates a vector from a human-readable string.
- static Vector3f FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes vector from a byte array.
- static float Dot (Vector3f lhs, Vector3f rhs)

  Calculates a dot product of two vectors.
- static Vector3f Cross (Vector3f lhs, Vector3f rhs)

  Calculates a cross product of two vectors.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Vector3f_GetRawBytesSize()

  A constant indicating the size of raw bytes of the vector.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- float x `[get, set]`

  An x coordinate.
- float y `[get, set]`

  A y coordinate.
- float z `[get, set]`

  A z coordinate.
- float this[int i] `[get, set]`

  Accesses vector dimension value via index.
- IntPtr nativeVectorObject `[get]`

  A getter of the native vector object.

Additional Inherited Members

7.35.1 Detailed Description

A 3-dimensional vector with single-precision floating-point values.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 Vector3f() `[1/3]`

```
MVCommon.Vector3f.Vector3f (
            float x,
            float y,
            float z )
```

A constructor.

Parameters

x	an x coordinate
y	a y coordinate
z	a z coordinate

7.35.2.2 Vector3f() [2/3]

```
MVCommon.Vector3f.Vector3f (
            Vector2f vector2,
            float z = 0.0f )
```

A constructor.

Parameters

vector2	a 2-dimensional vector whose x and y coordinates will be grabbed
z	a z coordinate

7.35.2.3 Vector3f() [3/3]

```
MVCommon.Vector3f.Vector3f (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native vector object

7.35.3 Member Function Documentation

7.35.3.1 Abs()

```
Vector3f MVCommon.Vector3f.Abs ( )
```

Creates a vector with dimensions with absolute values of the original vector.

Returns

a vector with absolute-valued dimensions

7.35.3.2 Clone()

```
Vector3f MVCommon.Vector3f.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.35.3.3 Cross()

```
static Vector3f MVCommon.Vector3f.Cross (
            Vector3f lhs,
            Vector3f rhs )  [static]
```

Calculates a cross product of two vectors.

Parameters

lhs	a left-hand-side vector-operand
rhs	a right-hand-side vector-operand

Returns

a vector representing the cross product

7.35.3.4 Dot()

```
static float MVCommon.Vector3f.Dot (
            Vector3f lhs,
            Vector3f rhs )  [static]
```

Calculates a dot product of two vectors.

Parameters

lhs	a first vector-operand
rhs	a second vector-operand

Returns

a dot product

7.35.3.5 FromRawBytes()

```
static Vector3f MVCommon.Vector3f.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes vector from a byte array.

Parameters

bytes	an array of vector bytes
consumeBytes	determines whether bytes of the vector shall be removed from the array

Returns

a vector

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.35.3.6 FromString()

```
static Vector3f MVCommon.Vector3f.FromString (
            String str )  [static]
```

Creates a vector from a human-readable string.

Parameters

str	a vector string

Returns

a vector

7.35.3.7 GetXY()

```
Vector2f MVCommon.Vector3f.GetXY ( )
```

Extracts x and y coordinates as a 2-dimensional vector.

Returns

a 2-dimensional vector

7.35.3.8 Inverted()

```
Vector3f MVCommon.Vector3f.Inverted ( )
```

Creates a vector with inverted dimensions (1/x).

Returns

an inverted vector

7.35.3.9 Length()

```
float MVCommon.Vector3f.Length ( )
```

Gets a length of the vector.

Returns

vector's length

7.35.3.10 Normalized()

```
Vector3f MVCommon.Vector3f.Normalized ( )
```

Creates a normalized vector (with length equal to 1).

Returns an unchanged vector in case its length is equal to 0.

Returns

a normalized vector

7.35.3.11 ToCommonString()

```
String MVCommon.Vector3f.ToCommonString ( )
```

Converts the vector into a human-readable string.

Returns

the vector string

7.35.3.12 ToRawBytes()

```
void MVCommon.Vector3f.ToRawBytes (
            ByteArray bytes )
```

Serializes the vector into a byte array.

Parameters

bytes	a byte array to serialize into

7.35.4 Property Documentation

7.35.4.1 this[int i]

```
float MVCommon.Vector3f.this[int i]  [get], [set]
```

Accesses vector dimension value via index.

Parameters

i	an index of the dimension to access

Returns

a dimension value

Exceptions

System.IndexOutOfRangeException	raised when index is out of range (0-2)

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

- public/math/Vector3f.cs

7.36 MVCommon.Vector4d Class Reference

A 4-dimensional vector with double-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Vector4d >.

Public Member Functions

- Vector4d ()
  A default constructor.
- Vector4d (double x, double y, double z, double w)
  A constructor.
- Vector4d (Vector3d vector3, double w=0.0)
  A constructor.
- Vector4d (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the vector into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the vector into a byte array.
- double Length ()
  Gets a length of the vector.
- Vector4d Inverted ()
  Creates a vector with inverted dimensions (1/x).
- Vector4d Normalized ()
  Creates a normalized vector (with length equal to 1).
- Vector4d Abs ()
  Creates a vector with dimensions with absolute values of the original vector.
- Vector3d GetXYZ ()
  Extracts x, y and z coordinates as a 3-dimensional vector.
- Vector4d Clone ()
  Makes an independent clone object.

Static Public Member Functions

- static Vector4d FromString (String str)
  Creates a vector from a human-readable string.
- static Vector4d FromRawBytes (ByteArray bytes, bool consumeBytes=false)
  Deserializes vector from a byte array.
- static double Dot (Vector4d lhs, Vector4d rhs)
  Calculates a dot product of two vectors.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Vector4d_GetRawBytesSize()
  A constant indicating the size of raw bytes of the vector.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- double x `[get, set]`

  An x coordinate.
- double y `[get, set]`

  A y coordinate.
- double z `[get, set]`

  A z coordinate.
- double w `[get, set]`

  A w coordinate.
- double this[int i] `[get, set]`

  Accesses vector dimension value via index.
- IntPtr nativeVectorObject `[get]`

  A getter of the native vector object.

Additional Inherited Members

7.36.1 Detailed Description

A 4-dimensional vector with double-precision floating-point values.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 Vector4d() `[1/3]`

```
MVCommon.Vector4d.Vector4d (
            double x,
            double y,
            double z,
            double w )
```

A constructor.

Parameters

x	an x coordinate
y	a y coordinate
z	a z coordinate
w	a w coordinate

7.36.2.2 Vector4d() [2/3]

```
MVCommon.Vector4d.Vector4d (
            Vector3d vector3,
            double w = 0.0 )
```

A constructor.

Parameters

vector3	a 3-dimensional vector whose x, y and z coordinates will be grabbed
w	a w coordinate

7.36.2.3 Vector4d() [3/3]

```
MVCommon.Vector4d.Vector4d (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native vector object

7.36.3 Member Function Documentation

7.36.3.1 Abs()

```
Vector4d MVCommon.Vector4d.Abs ( )
```

Creates a vector with dimensions with absolute values of the original vector.

Returns

a vector with absolute-valued dimensions

7.36.3.2 Clone()

```
Vector4d MVCommon.Vector4d.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.36.3.3 Dot()

```
static double MVCommon.Vector4d.Dot (
            Vector4d lhs,
            Vector4d rhs )  [static]
```

Calculates a dot product of two vectors.

Parameters

lhs	a first vector-operand
rhs	a second vector-operand

Returns

a dot product

7.36.3.4 FromRawBytes()

```
static Vector4d MVCommon.Vector4d.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes vector from a byte array.

Parameters

bytes	an array of vector bytes
consumeBytes	determines whether bytes of the vector shall be removed from the array

Returns

a vector

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.36.3.5 FromString()

```
static Vector4d MVCommon.Vector4d.FromString (
            String str )  [static]
```

Creates a vector from a human-readable string.

Parameters

str	a vector string

Returns

a vector

7.36.3.6 GetXYZ()

```
Vector3d MVCommon.Vector4d.GetXYZ ( )
```

Extracts x, y and z coordinates as a 3-dimensional vector.

Returns

a 3-dimensional vector

7.36.3.7 Inverted()

```
Vector4d MVCommon.Vector4d.Inverted ( )
```

Creates a vector with inverted dimensions (1/x).

Returns

an inverted vector

7.36.3.8 Length()

```
double MVCommon.Vector4d.Length ( )
```

Gets a length of the vector.

Returns

vector's length

7.36.3.9 Normalized()

```
Vector4d MVCommon.Vector4d.Normalized ( )
```

Creates a normalized vector (with length equal to 1).

Returns an unchanged vector in case its length is equal to 0.

Returns

a normalized vector

7.36.3.10 ToCommonString()

```
String MVCommon.Vector4d.ToCommonString ( )
```

Converts the vector into a human-readable string.

Returns

the vector string

7.36.3.11 ToRawBytes()

```
void MVCommon.Vector4d.ToRawBytes (
            ByteArray bytes )
```

Serializes the vector into a byte array.

Parameters

bytes	a byte array to serialize into

7.36.4 Property Documentation

7.36.4.1 this[int i]

```
double MVCommon.Vector4d.this[int i]  [get], [set]
```

Accesses vector dimension value via index.

Parameters

i	an index of the dimension to access

Returns

a dimension value

Exceptions

System.IndexOutOfRangeException	raised when index is out of range (0-3)

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

- public/math/Vector4d.cs

7.37 MVCommon.Vector4f Class Reference

A 4-dimensional vector with single-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Vector4f >.

Public Member Functions

- Vector4f ()
  A default constructor.
- Vector4f (float x, float y, float z, float w)
  A constructor.
- Vector4f (Vector3f vector3, float w=0.0f)
  A constructor.
- Vector4f (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the vector into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the vector into a byte array.

- float Length ()

  Gets a length of the vector.
- Vector4f Inverted ()

  Creates a vector with inverted dimensions (1/x).
- Vector4f Normalized ()

  Creates a normalized vector (with length equal to 1).
- Vector4f Abs ()

  Creates a vector with dimensions with absolute values of the original vector.
- Vector3f GetXYZ ()

  Extracts x, y and z coordinates as a 3-dimensional vector.
- Vector4f Clone ()

  Makes an independent clone object.

Static Public Member Functions

- static Vector4f FromString (String str)

  Creates a vector from a human-readable string.
- static Vector4f FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes vector from a byte array.
- static float Dot (Vector4f lhs, Vector4f rhs)

  Calculates a dot product of two vectors.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Vector4f_GetRawBytesSize()

  A constant indicating the size of raw bytes of the vector.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- float x `[get, set]`

  An x coordinate.
- float y `[get, set]`

  A y coordinate.
- float z `[get, set]`

  A z coordinate.
- float w `[get, set]`

  A w coordinate.
- float this[int i] `[get, set]`

  Accesses vector dimension value via index.
- IntPtr nativeVectorObject `[get]`

  A getter of the native vector object.

Additional Inherited Members

7.37.1 Detailed Description

A 4-dimensional vector with single-precision floating-point values.

7.37.2 Constructor & Destructor Documentation

7.37.2.1 Vector4f() [1/3]

```
MVCommon.Vector4f.Vector4f (
            float x,
            float y,
            float z,
            float w )
```

A constructor.

Parameters

x	an x coordinate
y	a y coordinate
z	a z coordinate
w	a w coordinate

7.37.2.2 Vector4f() [2/3]

```
MVCommon.Vector4f.Vector4f (
            Vector3f vector3,
            float w = 0.0f )
```

A constructor.

Parameters

vector3	a 3-dimensional vector whose x, y and z coordinates will be grabbed
w	a w coordinate

7.37.2.3 Vector4f() [3/3]

```
MVCommon.Vector4f.Vector4f (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native vector object

7.37.3 Member Function Documentation

7.37.3.1 Abs()

```
Vector4f MVCommon.Vector4f.Abs ( )
```

Creates a vector with dimensions with absolute values of the original vector.

Returns

a vector with absolute-valued dimensions

7.37.3.2 Clone()

```
Vector4f MVCommon.Vector4f.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.37.3.3 Dot()

```
static float MVCommon.Vector4f.Dot (
            Vector4f lhs,
            Vector4f rhs )  [static]
```

Calculates a dot product of two vectors.

Parameters

lhs	a first vector-operand
rhs	a second vector-operand

Returns

a dot product

7.37.3.4 FromRawBytes()

```
static Vector4f MVCommon.Vector4f.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes vector from a byte array.

Parameters

bytes	an array of vector bytes
consumeBytes	determines whether bytes of the vector shall be removed from the array

Returns

a vector

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array

7.37.3.5 FromString()

```
static Vector4f MVCommon.Vector4f.FromString (
            String str )  [static]
```

Creates a vector from a human-readable string.

Parameters

str	a vector string

Returns

a vector

7.37.3.6 GetXYZ()

```
Vector3f MVCommon.Vector4f.GetXYZ ( )
```

Extracts x, y and z coordinates as a 3-dimensional vector.

Returns

a 3-dimensional vector

7.37.3.7 Inverted()

```
Vector4f MVCommon.Vector4f.Inverted ( )
```

Creates a vector with inverted dimensions (1/x).

Returns

an inverted vector

7.37.3.8 Length()

```
float MVCommon.Vector4f.Length ( )
```

Gets a length of the vector.

Returns

vector's length

7.37.3.9 Normalized()

```
Vector4f MVCommon.Vector4f.Normalized ( )
```

Creates a normalized vector (with length equal to 1).

Returns an unchanged vector in case its length is equal to 0.

Returns

a normalized vector

7.37.3.10 ToCommonString()

```
String MVCommon.Vector4f.ToCommonString ( )
```

Converts the vector into a human-readable string.

Returns

the vector string

7.37.3.11 ToRawBytes()

```
void MVCommon.Vector4f.ToRawBytes (
            ByteArray bytes )
```

Serializes the vector into a byte array.

Parameters

bytes	a byte array to serialize into

7.37.4 Property Documentation

7.37.4.1 this[int i]

```
float MVCommon.Vector4f.this[int i]  [get], [set]
```

Accesses vector dimension value via index.

Parameters

i	an index of the dimension to access

Returns

a dimension value

Exceptions

System.IndexOutOfRangeException	raised when index is out of range (0-3)

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

- public/math/Vector4f.cs

7.38 MVCommon.VersionInfo Class Reference

A structure holding module version information.

Inherits MVCommon.NativeObjectHolder, and IEquatable< VersionInfo >.

Public Member Functions

- VersionInfo ()
  A constructor.
- VersionInfo (UInt32 major=0, UInt32 minor=0, UInt32 patch=0)
  A constructor.
- VersionInfo (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the version info into a string with format 'major.minor.patch'.

Protected Member Functions

- override void DestroyNativeObject ()
  Destroys the native object in a customized way.

Properties

- UInt32 major `[get, set]`
  Most-significant version component.
- UInt32 minor `[get, set]`
  Medium-significant version component.
- UInt32 patch `[get, set]`
  Least-significant version component.

Additional Inherited Members

7.38.1 Detailed Description

A structure holding module version information.

7.38.2 Constructor & Destructor Documentation

7.38.2.1 VersionInfo() [1/2]

```
MVCommon.VersionInfo.VersionInfo (
            UInt32 major = 0,
            UInt32 minor = 0,
            UInt32 patch = 0 )
```

A constructor.

Parameters

major	most-significant version component
minor	medium-significant version component
patch	least-significant version component

7.38.2.2 VersionInfo() [2/2]

```
MVCommon.VersionInfo.VersionInfo (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native VersionInfo object

7.38.3 Member Function Documentation

7.38.3.1 ToCommonString()

```
String MVCommon.VersionInfo.ToCommonString ( )
```

Converts the version info into a string with format 'major.minor.patch'.

Returns

a string containing version

7.38.4 Property Documentation

7.38.4.1 major

```
UInt32 MVCommon.VersionInfo.major  [get], [set]
```

Most-significant version component.

Difference indicates binary-incompatibility.

7.38.4.2 minor

```
UInt32 MVCommon.VersionInfo.minor  [get], [set]
```

Medium-significant version component.

Increased whenever a new official version is released.

7.38.4.3 patch

```
UInt32 MVCommon.VersionInfo.patch  [get], [set]
```

Least-significant version component.

Increased whenever an officially released version is patched and re-released.

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

- public/utils/VersionInfo.cs

7.39 MVCommon.Versord Class Reference

A rotational quaternion (i.e. versor) with double-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Versord >.

Public Member Functions

- Versord ()
  A constructor of an identity versor (i.e. no rotation).
- Versord (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the versor into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the versor into a byte array.
- Vector4d ToElementsVector ()
  Converts the versor into a vector with values of versor's internal elements.
- double[ ] ToRawElements ()
  Serializes the versor into an elements array.
- Versord Inverted ()
  Creates an inverted versor.
- Vector3d ToEulerAnglesZYX ()
  Converts the versor to Euler angles (in degrees) in z -> y -> x order.
- Versord Clone ()
  Makes an independent clone object.

Static Public Member Functions

- static Versord FromString (String str)

  Creates a versor from a human-readable string.
- static Versord FromRawBytes (ByteArray bytes, bool consumeBytes=false)

  Deserializes versor from a byte array.
- static Versord FromElementsVector (Vector4d elements)

  Creates a versor from a vector with values of versor's internal elements.
- static Versord FromRawElements (double[ ] elements)

  Deserializes versor from an elements array.
- static Versord CreateRotationAroundAxis (Vector3d axis, double angle)

  Creates a versor from axis of rotation and an angle (in degrees).
- static Versord CreateRotationFromMatrix (Matrix4x4d matrix)

  Creates a versor from a rotational part of transformation matrix.
- static Versord CreateRotationFromEulerAnglesZYX (Vector3d eulerAngles)

  Creates a versor from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Versord_GetRawBytesSize()

  A constant indicating the size of raw bytes of the versor.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeVersorObject `[get]`

  A getter of the native versor object.

Additional Inherited Members

7.39.1 Detailed Description

A rotational quaternion (i.e. versor) with double-precision floating-point values.

7.39.2 Constructor & Destructor Documentation

7.39.2.1 Versord()

```
MVCommon.Versord.Versord (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native versor object

7.39.3 Member Function Documentation

7.39.3.1 Clone()

```
Versord MVCommon.Versord.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.39.3.2 CreateRotationAroundAxis()

```
static Versord MVCommon.Versord.CreateRotationAroundAxis (
            Vector3d axis,
            double angle )  [static]
```

Creates a versor from axis of rotation and an angle (in degrees).

Parameters

axis	an axis of rotation
angle	an angle

Returns

a versor

7.39.3.3 CreateRotationFromEulerAnglesZYX()

```
static Versord MVCommon.Versord.CreateRotationFromEulerAnglesZYX (
            Vector3d eulerAngles )  [static]
```

Creates a versor from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.

Parameters

eulerAngles	Euler angles of Z-Y-X rotation

Returns

a versor

7.39.3.4 CreateRotationFromMatrix()

```
static Versord MVCommon.Versord.CreateRotationFromMatrix (
            Matrix4x4d matrix )  [static]
```

Creates a versor from a rotational part of transformation matrix.

Parameters

matrix	a matrix to extract the rotation from

Returns

a versor

7.39.3.5 FromElementsVector()

```
static Versord MVCommon.Versord.FromElementsVector (
            Vector4d elements )  [static]
```

Creates a versor from a vector with values of versor's internal elements.

Parameters

elements	a vector with versor's internal elements

Returns

a versor

Exceptions

System.ArgumentException	raied when the vector does not represent a rotational quaternion

7.39.3.6 FromRawBytes()

```
static Versord MVCommon.Versord.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes versor from a byte array.

Parameters

bytes	an array of versor bytes
consumeBytes	determines whether bytes of the versor shall be removed from the array

Returns

a versor

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array or the bytes do not represent a rotational quaternion

7.39.3.7 FromRawElements()

```
static Versord MVCommon.Versord.FromRawElements (
            double[] elements )  [static]
```

Deserializes versor from an elements array.

Parameters

elements	an array of 4 elements

Returns

a versor

Exceptions

System.ArgumentException	raised when the elements array has less than 4 elements or the elements do not represent a rotational quaternion

7.39.3.8 FromString()

```
static Versord MVCommon.Versord.FromString (
            String str )  [static]
```

Creates a versor from a human-readable string.

Parameters

str	a versor string

Returns

a versor

Exceptions

System.ArgumentException	raied when the string does not represent a rotational quaternion

7.39.3.9 Inverted()

```
Versord MVCommon.Versord.Inverted ( )
```

Creates an inverted versor.

Returns

an inverted versor

7.39.3.10 ToCommonString()

```
String MVCommon.Versord.ToCommonString ( )
```

Converts the versor into a human-readable string.

Returns

the versor string

7.39.3.11 ToElementsVector()

```
Vector4d MVCommon.Versord.ToElementsVector ( )
```

Converts the versor into a vector with values of versor's internal elements.

Returns

a vector of versor's elements

7.39.3.12 ToEulerAnglesZYX()

```
Vector3d MVCommon.Versord.ToEulerAnglesZYX ( )
```

Converts the versor to Euler angles (in degrees) in z -> y -> x order.

Returns

Euler angles of Z-Y-X rotation

7.39.3.13 ToRawBytes()

```
void MVCommon.Versord.ToRawBytes (
            ByteArray bytes )
```

Serializes the versor into a byte array.

Parameters

bytes	a byte array to serialize into

7.39.3.14 ToRawElements()

```
double [] MVCommon.Versord.ToRawElements ( )
```

Serializes the versor into an elements array.

Returns

an array of 4 elements

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

- public/math/Versord.cs

7.40 MVCommon.Versorf Class Reference

A rotational quaternion (i.e. versor) with single-precision floating-point values.

Inherits MVCommon.NativeObjectHolder, and IEquatable< Versorf >.

Public Member Functions

- Versorf ()
  A constructor of an identity versor (i.e. no rotation).
- Versorf (IntPtr nativeObject)
  A constructor.
- String ToCommonString ()
  Converts the versor into a human-readable string.
- void ToRawBytes (ByteArray bytes)
  Serializes the versor into a byte array.
- Vector4f ToElementsVector ()
  Converts the versor into a vector with values of versor's internal elements.
- float[ ] ToRawElements ()
  Serializes the versor into an elements array.
- Versorf Inverted ()
  Creates an inverted versor.
- Vector3f ToEulerAnglesZYX ()
  Converts the versor to Euler angles (in degrees) in z -> y -> x order.
- Versorf Clone ()
  Makes an independent clone object.

Static Public Member Functions

- static Versorf FromString (String str)
  Creates a versor from a human-readable string.
- static Versorf FromRawBytes (ByteArray bytes, bool consumeBytes=false)
  Deserializes versor from a byte array.
- static Versorf FromElementsVector (Vector4f elements)
  Creates a versor from a vector with values of versor's internal elements.
- static Versorf FromRawElements (float[ ] elements)
  Deserializes versor from an elements array.
- static Versorf CreateRotationAroundAxis (Vector3f axis, float angle)
  Creates a versor from axis of rotation and an angle (in degrees).
- static Versorf CreateRotationFromMatrix (Matrix4x4f matrix)
  Creates a versor from a rotational part of transformation matrix.
- static Versorf CreateRotationFromEulerAnglesZYX (Vector3f eulerAngles)
  Creates a versor from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.

Static Public Attributes

- static readonly UInt64 RAW_BYTES_SIZE = Versorf_GetRawBytesSize()
  A constant indicating the size of raw bytes of the versor.

Protected Member Functions

- override void DestroyNativeObject ()

  Destroys the native object in a customized way.

Properties

- IntPtr nativeVersorObject `[get]`

  A getter of the native versor object.

Additional Inherited Members

7.40.1 Detailed Description

A rotational quaternion (i.e. versor) with single-precision floating-point values.

7.40.2 Constructor & Destructor Documentation

7.40.2.1 Versorf()

```
MVCommon.Versorf.Versorf (
            IntPtr nativeObject )
```

A constructor.

Parameters

nativeObject	a native versor object

7.40.3 Member Function Documentation

7.40.3.1 Clone()

```
Versorf MVCommon.Versorf.Clone ( )
```

Makes an independent clone object.

The clone's values are copied from this object.

Returns

a clone object

7.40.3.2 CreateRotationAroundAxis()

```
static Versorf MVCommon.Versorf.CreateRotationAroundAxis (
            Vector3f axis,
            float angle )  [static]
```

Creates a versor from axis of rotation and an angle (in degrees).

Parameters

axis	an axis of rotation
angle	an angle

Returns

a versor

7.40.3.3 CreateRotationFromEulerAnglesZYX()

```
static Versorf MVCommon.Versorf.CreateRotationFromEulerAnglesZYX (
            Vector3f eulerAngles )  [static]
```

Creates a versor from Euler angles (in degrees) in eulerAngles.z -> eulerAngles.y -> eulerAngles.x order.

Parameters

eulerAngles	Euler angles of Z-Y-X rotation

Returns

a versor

7.40.3.4 CreateRotationFromMatrix()

```
static Versorf MVCommon.Versorf.CreateRotationFromMatrix (
            Matrix4x4f matrix )  [static]
```

Creates a versor from a rotational part of transformation matrix.

Parameters

matrix	a matrix to extract the rotation from

Returns

a versor

7.40.3.5 FromElementsVector()

```
static Versorf MVCommon.Versorf.FromElementsVector (
            Vector4f elements )  [static]
```

Creates a versor from a vector with values of versor's internal elements.

Parameters

elements	a vector with versor's internal elements

Returns

a versor

Exceptions

System.ArgumentException	raied when the vector does not represent a rotational quaternion

7.40.3.6 FromRawBytes()

```
static Versorf MVCommon.Versorf.FromRawBytes (
            ByteArray bytes,
            bool consumeBytes = false )  [static]
```

Deserializes versor from a byte array.

Parameters

bytes	an array of versor bytes
consumeBytes	determines whether bytes of the versor shall be removed from the array

Returns

a versor

Exceptions

System.ArgumentException	raised when there are not enough bytes in the array or the bytes do not represent a rotational quaternion

7.40.3.7 FromRawElements()

```
static Versorf MVCommon.Versorf.FromRawElements (
            float[] elements )  [static]
```

Deserializes versor from an elements array.

Parameters

elements	an array of 4 elements

Returns

a versor

Exceptions

System.ArgumentException	raised when the elements array has less than 4 elements or the elements do not represent a rotational quaternion

7.40.3.8 FromString()

```
static Versorf MVCommon.Versorf.FromString (
            String str )  [static]
```

Creates a versor from a human-readable string.

Parameters

str	a versor string

Returns

a versor

Exceptions

System.ArgumentException	raied when the string does not represent a rotational quaternion

7.40.3.9 Inverted()

```
Versorf MVCommon.Versorf.Inverted ( )
```

Creates an inverted versor.

Returns

an inverted versor

7.40.3.10 ToCommonString()

```
String MVCommon.Versorf.ToCommonString ( )
```

Converts the versor into a human-readable string.

Returns

the versor string

7.40.3.11 ToElementsVector()

```
Vector4f MVCommon.Versorf.ToElementsVector ( )
```

Converts the versor into a vector with values of versor's internal elements.

Returns

a vector of versor's elements

7.40.3.12 ToEulerAnglesZYX()

```
Vector3f MVCommon.Versorf.ToEulerAnglesZYX ( )
```

Converts the versor to Euler angles (in degrees) in z -> y -> x order.

Returns

Euler angles of Z-Y-X rotation

7.40.3.13 ToRawBytes()

```
void MVCommon.Versorf.ToRawBytes (
            ByteArray bytes )
```

Serializes the versor into a byte array.

Parameters

bytes	a byte array to serialize into

7.40.3.14 ToRawElements()

```
float [] MVCommon.Versorf.ToRawElements ( )
```

Serializes the versor into an elements array.

Returns

an array of 4 elements

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

- public/math/Versorf.cs

Index