Quinn Math Library Documentation

Headers:

“QuinnMathLibAll.h”
- contains an include to all other headers

“General.h”
- contains the General class with static functions

“Vector2.h”
- contains the Vector2 class

“Vector3.h”
- contains the Vector3 class

“Vector4.h”
- contains the Vector4 class

“Matrix3.h”
- contains the Matrix3 class

“Matrix4.h”
- contains the Matrix4 class

Classes:

General
- a static class that contains static functions that don’t relate to any class

Vector2
- representation of a 2D vector
- fields: float x, float y,

Vector3
- representation of a 3D vector
- fields: float x, float y, float z

Vector4
- representation of a color
- fields: float w, float x, float y, float z

Matrix3
- a 2D translation matrix
- fields: float matrix[3][3]

Matrix4
- a 3D translation matrix
- fields: float matrix[4][]

Functions:

General::

static float Lerp(float in_begin, float in_end, float in_percent)
- Returns a scalar that is in_percent(0 - 1)% between in_begin and in_end

static float ToDegrees(float in_radians)
- returns the Degree equivalent of in_radians

Functions cont’d:

General::(cont’d)

static float ToRadians(float in_Degrees)

-returns the Radian equivalent of in_Degrees

static float ShiftPowOfTwo(float in_scalar);

-returns the closest power of two to in_scalar

Vector2::

Vector2()

-constructs a Vector2 with the fields x and y equal to 0

Vector2(float in_x, float in_y)

-constructs a Vector2 with field x = in_x and field y = in_y

float DotProduct(Vector2 other)

-returns the dot product of the instance called upon and the given Vector2

static float DotProduct(Vector2 in_a, Vector2 in_b)

-returns the dot product of the two given Vector2s

Vector2 CrossProduct(Vector2 other)

-returns a Vector2 containing the cross product of the instance called upon and the given Vector2

static Vector2 CrossProduct(Vector2 in_a, Vector2 in_b)

-returns a Vector2 containing the cross product containing the two given Vector2s

float Magnitude()

-returns the magnitude of the instance called upon

static float Magnitude(Vector2 input)

-returns the magnitude of the given Vector2

Vector2 Normalize()

-returns a normalized version of the instance called upon

static Vector2 Normalize(Vector2 input)

-returns a normalized version of the Vector2 given

void NormalizeThis()

-sets the instance called upon to a normalized version of itself

Vector2 Lerp(Vector2 other, float in_percent)

-returns a Vector2 that is in_percent(0 - 1) between the instance called upon and the Vector2 given

static Vector2 Lerp(Vector2 in_first, Vector2 in_second, float in_percent)

-returns a Vector2 that is in_percent(0 - 1) between the Vector2s that are given

Vector3::

Vector3()

-constructs a Vector3 with the fields x, y and z equal to 0

Vector3(float in_x, float in_y, float in_z)

-constructs a Vector3 with field x = in_x, field y = in_y and field z = in_z

Functions cont’d:

Vector3::(cont’d)

float DotProduct(Vector3 other)

-returns the dot product of the instance called upon and the given Vector3

static float DotProduct(Vector3 in_a, Vector3 in_b)

-returns the dot product of the two given Vector3s

Vector3 CrossProduct(Vector3 other)

-returns a Vector3 containing the cross product of the instance called upon and the given Vector3

static Vector3 CrossProduct(Vector3 in_a, Vector3 in_b)

-returns a Vector3 containing the cross product containing the two given Vector3s

float Magnitude()

-returns the magnitude of the instance called upon

static float Magnitude(Vector3 input)

-returns the magnitude of the given Vector3

Vector3 Normalize()

-returns a normalized version of the instance called upon

static Vector3 Normalize(Vector3 input)

-returns a normalized version of the Vector3 given

void NormalizeThis()

-sets the instance called upon to a normalized version of itself

Vector3 Lerp(Vector3 other, float in_percent)

-returns a Vector3 that is in_percent(0 - 1) between the instance called upon and the Vector3 given

static Vector3 Lerp(Vector3 in_first, Vector3 in_second, float in_percent)

-returns a Vector3 that is in_percent(0 - 1) between the Vector3s that are given

Vector4::

Vector4()

-constructs a Vector4 with the fields w, x, y and z equal to 0

Vector4(float in_x, float in_y, float in_z, float in_w)

-constructs a Vector4 with field x = in_x, field y = in_y, field z = in_z and field w = in_w

static Vector4 ConstructFromColor(float in_Alpha, float in_Red, float in_Green, float in_Blue)

-returns a Vector4 with field w = in_alpha / 250 field x = in_red / 250, field y = in_Green / 250 and field z = in_Blue / 250

static Vector4 ConstructFromColor(unsigned int in_hexColor)

-converts a hexadecimal number into a Vector4

float Magnitude()

-returns the magnitude of the instance called upon static float Magnitude(Vector4 input)

-returns the magnitud of the given Vector4

Functions cont’d:

Vector4::(cont’d)

Vector4 Normalize()

-returns a normalized version of the instance called upon

static Vector4 Normalize(Vector4 input)

-returns a normalized version of the vector4 given

Matrix3::

Matrix3()

-creates a Matrix3 in identity form

~Matrix3()

-destroys instantiated Matrix3

Matrix3 Rotation(float in_degrees)

-creates and sets a matrix for the given degrees

Matrix3 Scale(float in_xScale, float in_yScale)

-creates and sets a matrix to scale the given amount

Matrix3 TransformVector(float in_xTransform, float in_yTransform)

-creates and sets a matrix to transform a vector the given amount

Matrix3 Transpose()

-transposes the instance called upon

void Set(int in_col, int in_row, float in_value)

-sets the matrix at [in_col][in_row] to in_value

void Set(float in_00, float in_01, float in_02, float in_10, float in_11, float in_12, float in_20, float in_21, float in_22)

-sets the matrix to the given values

Matrix4::

Matrix4()

-creates a Matrix4 in identity form

~Matrix4()

-destroys instantiated Matrix4

Matrix4 XRotation(float in_degrees)

-creates a rotation matrix for the x axis

Matrix4 YRotation(float in_degrees)

-creates a rotation matrix for the y axis

Matrix4 ZRotation(float in_degrees)

-creates a rotation matrix for the z axis

Matrix4 Scale(float in_xScale, float in_yScale, float in_zScale)

-creates and sets a matrix to scale the given amount

Matrix4 TransformVector(float in_xTransform, float in_yTransform, float in_zTransform)

-creates and sets a matrix to transform a vector the given amount

Matrix4 Transpose()

  - -transposes the instance called upon
- void Set(int in_col, int in_row, float in_value)
  - -sets the matrix at [in_col][in_row] to in_value

Functions cont’d:

Matrix4::(cont’d)

- void Set(float in_00, float in_01, float in_02, float in_10, float in_11, float in_12, float in_20, float in_21, float in_22)
  - -sets the matrix to the given values
- Matrix4 OrthoProj(float in_top, float in_bottom, float in_right, float in_left, float in_far, float in_near)
  - -creates a matrix for Orthographic projection for the given values

Operators:

Vector2:

- Vector2 + Vector2 returns Vector2
  - adds each field to it’s counterpart
- Vector2 - Vector2 returns Vector2
  - subtracts each field from it’s counterpart
- Vector2 * Vector2 returns Vector2
  - multiplies each field by its counterpart
- Vector2 * float returns Vector2
  - multiplies each field by the given value
- Vector2 += Vector2 returns Void
  - adds each field to it’s counterpart and saves the value
- Vector2 -= Vector2 returns Void
  - subtracts each field from it’s counterpart and saves the value
- Vector2 *= Vector2 returns Void
  - multiplies each field by its counterpart and saves the value
- Vector2 *= float returns Void
  - multiplies each field by the given value and saves the value
- ostream << Vector2 returns ostream
  - outputs the value to the given ostream
- Vector2 == Vector2 returns bool
  - returns true if all values are within 0.00001 of each other
- Vector2 != Vector2 returns bool
  - returns the opposite of the equality operator

Vector3:

- Vector3 + Vector3 returns Vector3
  - adds each field to it’s counterpart
- Vector3 - Vector3 returns Vector3
  - subtracts each field from it’s counterpart
- Vector3 * Vector3 returns Vector3
  - multiplies each field by its counterpart

Vector3 * float returns Vector3
multiplies each field by the given value

Operators cont'd:

Vector3(cont'd):

Vector3 += Vector3 returns Void
adds each field to it's counterpart and saves the value

Vector3 -= Vector3 returns Void
subtracts each field from it's counterpart and saves the value

Vector3 *= Vector3 returns Void
multiplies each field by its counterpart and saves the value

Vector3 *= float returns Void
multiplies each field by the given value and saves the value

ostream << Vector3 returns ostream
outputs the value to the given ostream

Vector3 == Vector3 returns bool
returns true if all values are within 0.00001 of each other

Vector3 != Vector3 returns bool
returns the opposite of the equality operator

Vector4:

ostream << Vector4 returns ostream
outputs the value to the given ostream

Vector4 == Vector4 returns bool
returns true if all values are within 0.00001 of each other

Vector4 != Vector4 returns bool
returns the opposite of the equality operator

Matrix3:

Matrix3 + Matrix3 returns Matrix3
adds each value to its counterpart

Matrix3 - Matrix3 returns Matrix3
subtracts each field from it's counterpart

Matrix3 * Matrix3 returns Matrix3
performs matrix multiplication

Vector2 * Matrix3 returns Vector2
performs matrix multiplication (assuming 1 in the Z position of the Vector2)

Matrix3 += Matrix3 returns Matrix3
adds each field to it's counterpart and saves the value

Matrix3 -= Matrix3 returns Matrix3
subtracts each field from it's counterpart and saves the value

Matrix3 *= Matrix3 returns Matrix3
performs matrix multiplication and saves the value

ostream << Matrix3 returns ostream
outputs the value to the given ostream

Matrix3 == Matrix3 returns Matrix3

returns true if all values are within 0.00001 of each other

Operators cont'd:

Matrix3(cont'd):

Matrix3 != Matrix3 returns Matrix3

returns the opposite of the equality operator

Matrix4:

Matrix4 + Matrix3 returns Matrix4

adds each value to its counterpart

Matrix4 - Matrix4 returns Matrix4

subtracts each field from it's counterpart

Matrix4 * Matrix4 returns Matrix4

performs matrix multiplication

Vector3 * Matrix4 returns Vector3

performs matrix multiplication (assuming 1 in the W position of the Vector3)

Matrix4 += Matrix4 returns Matrix4

adds each field to it's counterpart and saves the value

Matrix4 -= Matrix4 returns Matrix4

subtracts each field from it's counterpart and saves the value

Matrix4 *= Matrix4 returns Matrix4

performs matrix multiplication and saves the value

ostream << Matrix4 returns ostream

outputs the value to the given ostream

Matrix4 == Matrix4 returns Matrix4

returns true if all values are within 0.00001 of each other

Matrix4 != Matrix4 returns Matrix4

returns the opposite of the equality operator