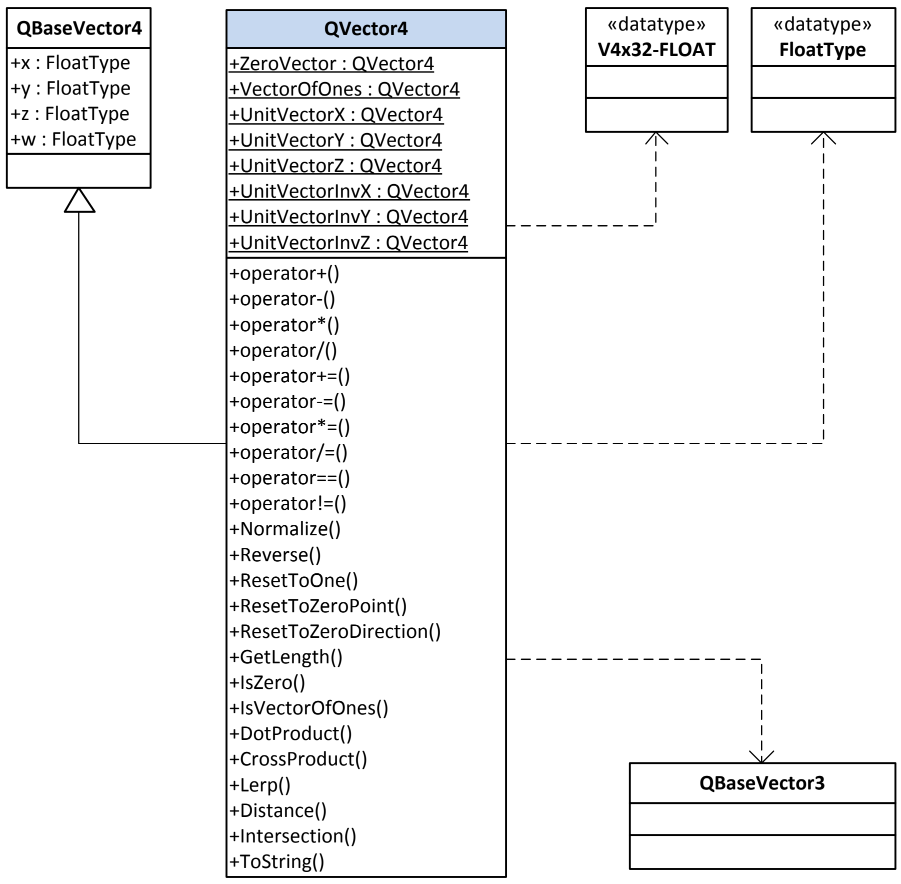
QVector4

Diagrams



Knowledge Requirements

- Math
- See: Introduction to 3D Game Programming with DirectX 9.0, Part I.
- See: *<Program Files Folder>\Microsoft DirectX SDK (June 2010)\Documentation\DirectX9\windows_graphics.chm*, from DirectX SDK. Search for D3DXVECTOR4 and D3DXVec4 in "Index" tab.
- See: *http://www.zator.com/Cpp/E4_9_18.htm* to refresh operators overloading knowledge.

Functional Specifications

- Override default constructor. Sets attributes to zero.
- Override copy constructor.
- Implement constructor that receives a QBaseVector4 type.
- Implement constructor that receives a QBaseVector3 type.
- Implement constructor that receives 4 FloatTypes, one for each vector components.
- Implement constructor that receives only 1 FloatType. Set all attributes to that value.
- Implement constructor that receives a 4-FloatTypes array.
- Implement constructor that receives a pointer-to-FloatType. The pointer should point to a dynamically allocated 4-FloatTypes array.

- Implement constructor that receives a V4x32-FLOAT.
- It is not necessary to override default destructor.
- It is not necessary to override assign operator.
- Operator* must offer an overload that receives a FloatType (product by scalar).
- Operator* must offer an overload that receives a QBaseVector4. Each vector component will be multiplied by the other vector's component.
- Operator/ must offer an overload that receives a FloatType (division by scalar).
- Operator/ must offer an overload that receives a QBaseVector4. Each vector component will be divided by the other vector's component.
- A global operator* must be implemented in order to let a FloatType be multiplied by a QVector4. It's not the same QVector4 * FloatType than FloatType * QVector4.
- A global operator/ must be implemented in order to let a FloatType be multiplied by a QVector4. It's not the same QVector4 / FloatType than FloatType / QVector4.
- ResetToOne sets all vector's components to 1.
- ResetToZeroPoint sets X, Y and Z components to 0, and W to 1.
- ResetToZeroDirection sets all vector's components to 0, including W.
- GetLength calculates vector module.
- When implementing CrossProduct, be aware of Quimera Engine uses a left-handed convention and that affects directly to the resulting vector.
- Lerp must receive a FloatType value between 0 and 1. When it receives 1, the vector keeps its length; when it receives 0.5, the vector length is reduced to its half, etc. If it receives 2, its length becomes double.
- Distance calculates the length of the difference between 2 points, in a homogeneous space. If W = 0 in any of two vectors, then the result is 0 (because they aren't points but directions).
- Intersection must offer an overload that receives a QBaseVector4 and returns a boolean type. If the vector intersects with the supplied vector, then return TRUE.
- Intersection must offer an overload that receives an input QBaseVector4, an output QBaseVector4 and returns a boolean type. If the vector intersects with the supplied input vector, then the output vector is filled with the intersection position and the method returns TRUE.
- ToString format: "V4(X, Y, Z, W)". Use STL string.

Design / Technical Requirements

- Use member initialization lists.
- Remember using "explicit" when constructors receive only one parameter.
- No virtual methods.
- Use by-reference parameters always.
- Operator== and Operator!= must have Epsilon value into account.
- Use FloatType constants to store values like 1.
- Try to avoid square roots.
- All methods should be inline.
- No exceptions.
- No error codes.
- No profiling.
- Check for division by zero. Use asserts.
- Respect diagram names.

Support People

- Thund.