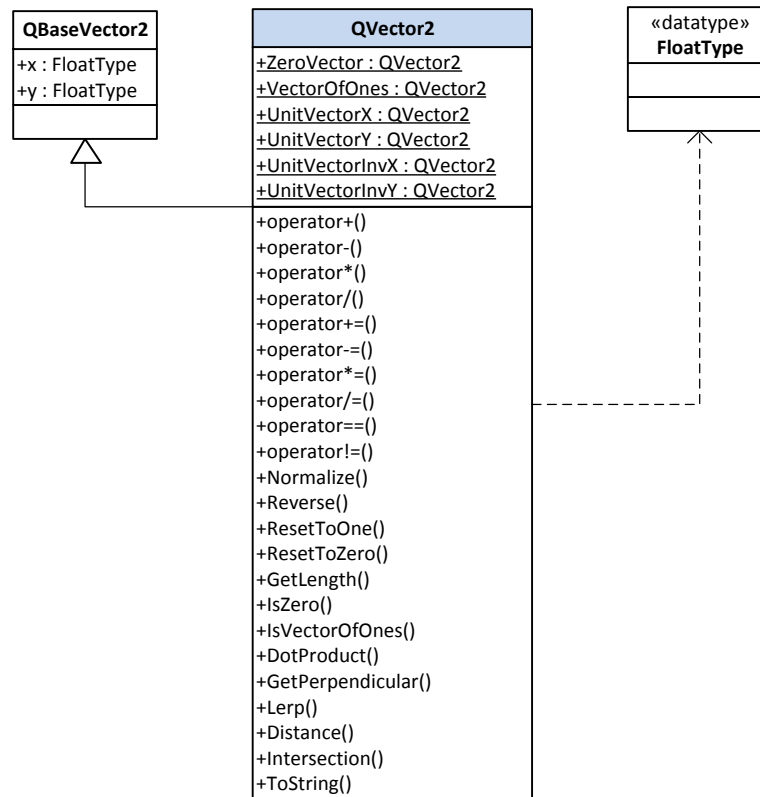


## Diagrams



## 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 D3DXVECTOR2 and D3DXVec2 in “Index” tab.
- See: [http://www.zator.com/Cpp/E4\\_9\\_18.htm](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 **QBaseVector2** type.
- Implement constructor that receives 2 **FloatType**s, one for each vector components.
- Implement constructor that receives only 1 **FloatType**. Set all attributes to that value.
- Implement constructor that receives a 2-**FloatType**s array.
- Implement constructor that receives a pointer-to-**FloatType**. The pointer should point to a dynamically allocated 2-**FloatType**s array.
- 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 QBaseVector2. 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 QBaseVector2. 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 QVector2. It's not the same QVector2 \* FloatType than FloatType \* QVector2.
- A global operator/ must be implemented in order to let a FloatType be multiplied by a QVector2. It's not the same QVector2 / FloatType than FloatType / QVector2.
- ResetToOne sets all vector's components to 1.
- GetLength calculates vector module.
- GetPerpendicular returns a unit vector which is perpendicular to the vector whose method is invoked. Remember that Quimera Engine uses a left-handed convention and that affects directly to the new vector direction.
- 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 vectors.
- Intersection must offer an overload that receives a QBaseVector2 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 QBaseVector2, an output QBaseVector2 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: "V2( X, Y)". 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.

