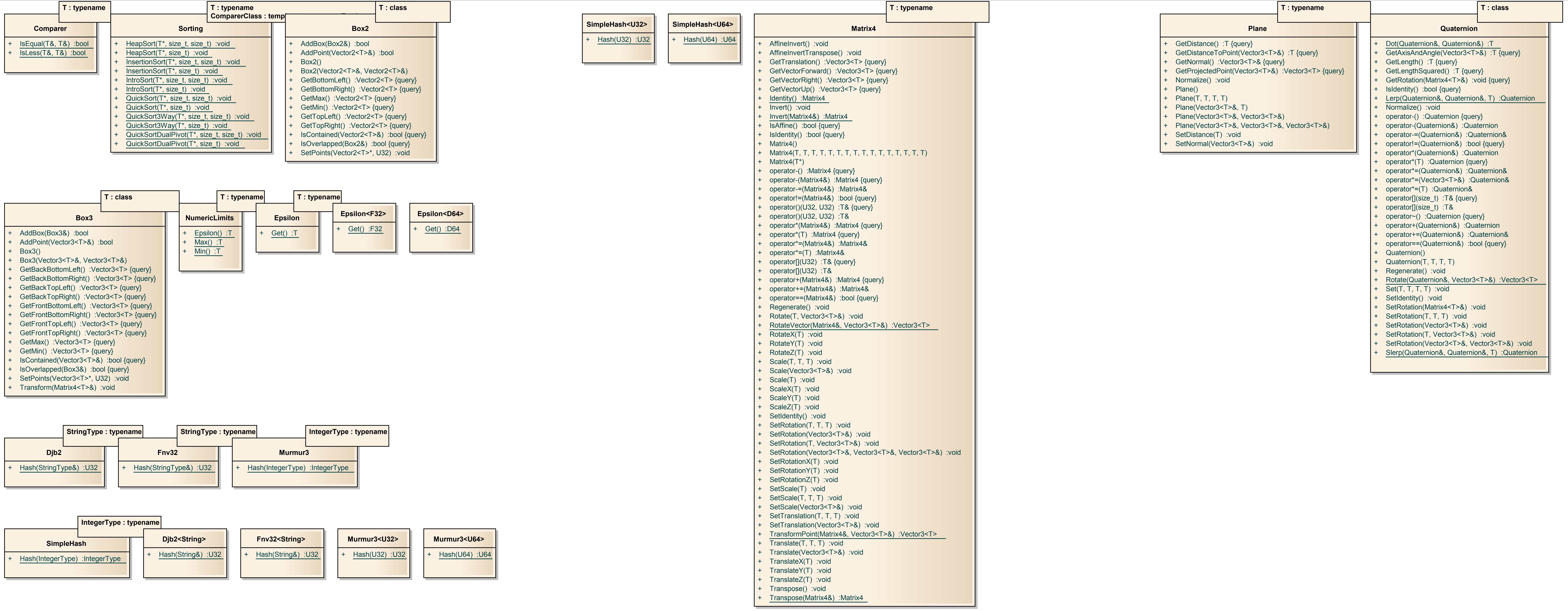
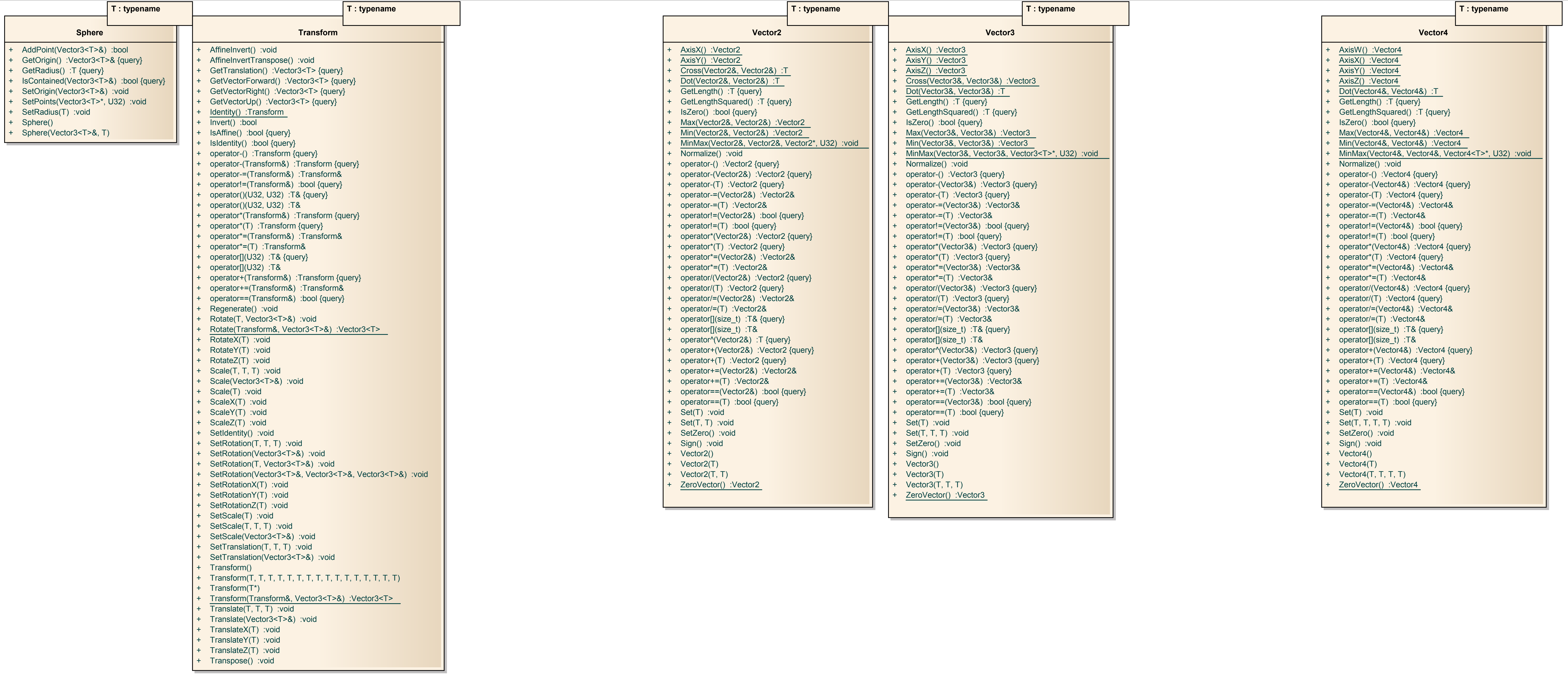


```
T : typename
Comparer
+  IsEqual(T&, T&)  :bool
+  IsLess(T&, T&)  :bool

T : typename
ComparerClass : temp
Sorting
+  HeapSort(T*, size_t, size_t)  :void
+  HeapSort(T*, size_t)  :void
+  InsertionSort(T*, size_t, size_t)  :void
+  InsertionSort(T*, size_t)  :void
+  IntroSort(T*, size_t, size_t)  :void
+  IntroSort(T*, size_t)  :void
+  QuickSort(T*, size_t, size_t)  :void
+  QuickSort(T*, size_t)  :void
+  QuickSort3Way(T*, size_t, size_t)  :void
+  QuickSort3Way(T*, size_t)  :void
+  QuickSortDualPivot(T*, size_t, size_t)  :void
+  QuickSortDualPivot(T*, size_t)  :void

T : class
Box2
+  AddBox(Box2&)  :bool
+  AddPoint(Vector2<T>&)  :bool
+  Box2()
+  Box2(Vector2<T>&, Vector2<T>&)
+  GetBottomLeft()  :Vector2<T> {query}
+  GetBottomRight()  :Vector2<T> {query}
+  GetMax()  :Vector2<T> {query}
+  GetMin()  :Vector2<T> {query}
+  GetTopLeft()  :Vector2<T> {query}
+  GetTopRight()  :Vector2<T> {query}
+  IsContained(Vector2<T>&)  :bool {query}
+  IsOverlapped(Box2&)  :bool {query}
+  SetPoints(Vector2<T>*, U32)  :void

SimpleHash<U32>
+  Hash(U32)  :U32

SimpleHash<U64>
+  Hash(U64)  :U64

T : typename
Matrix4
+  AffineInvert()  :void
+  AffineInvertTranspose()  :void
+  GetTranslation()  :Vector3<T> {query}
+  GetVectorForward()  :Vector3<T> {query}
+  GetVectorRight()  :Vector3<T> {query}
+  GetVectorUp()  :Vector3<T> {query}
+  Identity()  :Matrix4
+  Invert()  :void
+  Invert(Matrix4&)  :Matrix4
+  IsAffine()  :bool {query}
+  IsIdentity()  :bool {query}
+  Matrix4()
+  Matrix4(T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T)
+  Matrix4(T*)
+  operator-()  :Matrix4 {query}
+  operator-(Matrix4&)  :Matrix4 {query}
+  operator-=(Matrix4&)  :Matrix4&
+  operator!=(Matrix4&)  :bool {query}
+  operator()(U32, U32)  :T& {query}
+  operator()(U32, U32)  :T&
+  operator*(Matrix4&)  :Matrix4 {query}
+  operator*(T)  :Matrix4 {query}
+  operator*=(Matrix4&)  :Matrix4&
+  operator*=(T)  :Matrix4&
+  operator[](U32)  :T& {query}
+  operator[](U32)  :T&
+  operator+(Matrix4&)  :Matrix4 {query}
+  operator+=(Matrix4&)  :Matrix4&
+  operator==(Matrix4&)  :bool {query}
+  Regenerate()  :void
+  Rotate(T, Vector3<T>&)  :void
+  RotateVector(Matrix4&, Vector3<T>&)  :Vector3<T>
+  RotateX(T)  :void
+  RotateY(T)  :void
+  RotateZ(T)  :void
+  Scale(T, T, T)  :void
+  Scale(Vector3<T>&)  :void
+  Scale(T)  :void
+  ScaleX(T)  :void
+  ScaleY(T)  :void
+  ScaleZ(T)  :void
+  SetIdentity()  :void
+  SetRotation(T, T, T)  :void
+  SetRotation(Vector3<T>&)  :void
+  SetRotation(T, Vector3<T>&)  :void
+  SetRotation(Vector3<T>&, Vector3<T>&, Vector3<T>&)  :void
+  SetRotationX(T)  :void
+  SetRotationY(T)  :void
+  SetRotationZ(T)  :void
+  SetScale(T)  :void
+  SetScale(T, T, T)  :void
+  SetScale(Vector3<T>&)  :void
+  SetTranslation(T, T, T)  :void
+  SetTranslation(Vector3<T>&)  :void
+  TransformPoint(Matrix4&, Vector3<T>&)  :Vector3<T>
+  Translate(T, T, T)  :void
+  Translate(Vector3<T>&)  :void
+  TranslateX(T)  :void
+  TranslateY(T)  :void
+  TranslateZ(T)  :void
+  Transpose()  :void
+  Transpose(Matrix4&)  :Matrix4

T : typename
Plane
+  GetDistance()  :T {query}
+  GetDistanceToPoint(Vector3<T>&)  :T {query}
+  GetNormal()  :Vector3<T>& {query}
+  GetProjectedPoint(Vector3<T>&)  :Vector3<T> {query}
+  Normalize()  :void
+  Plane()
+  Plane(T, T, T, T)
+  Plane(Vector3<T>&, T)
+  Plane(Vector3<T>&, Vector3<T>&)
+  Plane(Vector3<T>&, Vector3<T>&, Vector3<T>&)
+  SetDistance(T)  :void
+  SetNormal(Vector3<T>&)  :void

T : class
Quaternion
+  Dot(Quaternion&, Quaternion&)  :T
+  GetAxisAndAngle(Vector3<T>&)  :T {query}
+  GetLength()  :T {query}
+  GetLengthSquared()  :T {query}
+  GetRotation(Matrix4<T>&)  :void {query}
+  IsIdentity()  :bool {query}
+  Lerp(Quaternion&, Quaternion&, T)  :Quaternion
+  Normalize()  :void
+  operator-()  :Quaternion {query}
+  operator-(Quaternion&)  :Quaternion
+  operator-=(Quaternion&)  :Quaternion&
+  operator!=(Quaternion&)  :bool {query}
+  operator*(Quaternion&)  :Quaternion
+  operator*(T)  :Quaternion {query}
+  operator*=(Quaternion&)  :Quaternion&
+  operator*=(Vector3<T>&)  :Quaternion&
+  operator*=(T)  :Quaternion&
+  operator[](size_t)  :T& {query}
+  operator[](size_t)  :T&
+  operator~()  :Quaternion {query}
+  operator+(Quaternion&)  :Quaternion
+  operator+=(Quaternion&)  :Quaternion&
+  operator==(Quaternion&)  :bool {query}
+  Quaternion()
+  Quaternion(T, T, T, T)
+  Regenerate()  :void
+  Rotate(Quaternion&, Vector3<T>&)  :Vector3<T>
+  Set(T, T, T, T)  :void
+  SetIdentity()  :void
+  SetRotation(Matrix4<T>&)  :void
+  SetRotation(T, T, T)  :void
+  SetRotation(Vector3<T>&)  :void
+  SetRotation(T, Vector3<T>&)  :void
+  SetRotation(Vector3<T>&, Vector3<T>&)  :void
+  Slerp(Quaternion&, Quaternion&, T)  :Quaternion

T : class
Box3
+  AddBox(Box3&)  :bool
+  AddPoint(Vector3<T>&)  :bool
+  Box3()
+  Box3(Vector3<T>&, Vector3<T>&)
+  GetBackBottomLeft()  :Vector3<T> {query}
+  GetBackBottomRight()  :Vector3<T> {query}
+  GetBackTopLeft()  :Vector3<T> {query}
+  GetBackTopRight()  :Vector3<T> {query}
+  GetFrontBottomLeft()  :Vector3<T> {query}
+  GetFrontBottomRight()  :Vector3<T> {query}
+  GetFrontTopLeft()  :Vector3<T> {query}
+  GetFrontTopRight()  :Vector3<T> {query}
+  GetMax()  :Vector3<T> {query}
+  GetMin()  :Vector3<T> {query}
+  IsContained(Vector3<T>&)  :bool {query}
+  IsOverlapped(Box3&)  :bool {query}
+  SetPoints(Vector3<T>*, U32)  :void
+  Transform(Matrix4<T>&)  :void

T : typename
NumericLimits
+  Epsilon()  :T
+  Max()  :T
+  Min()  :T

T : typename
Epsilon
+  Get()  :T

Epsilon<F32>
+  Get()  :F32

Epsilon<D64>
+  Get()  :D64

StringType : typename
Djb2
+  Hash(StringType&)  :U32

StringType : typename
Fnv32
+  Hash(StringType&)  :U32

IntegerType : typename
Murmur3
+  Hash(IntegerType)  :IntegerType

IntegerType : typename
SimpleHash
+  Hash(IntegerType)  :IntegerType

Djb2<String>
+  Hash(String&)  :U32

Fnv32<String>
+  Hash(String&)  :U32

Murmur3<U32>
+  Hash(U32)  :U32

Murmur3<U64>
+  Hash(U64)  :U64
```



```
T : typename
Sphere
+   AddPoint(Vector3<T>&)  :bool
+   GetOrigin()  :Vector3<T>& {query}
+   GetRadius()  :T {query}
+   IsContained(Vector3<T>&)  :bool {query}
+   SetOrigin(Vector3<T>&)  :void
+   SetPoints(Vector3<T>*, U32)  :void
+   SetRadius(T)  :void
+   Sphere()
+   Sphere(Vector3<T>&, T)
```

```
T : typename
Transform
+   AffineInvert()  :void
+   AffineInvertTranspose()  :void
+   GetTranslation()  :Vector3<T> {query}
+   GetVectorForward()  :Vector3<T> {query}
+   GetVectorRight()  :Vector3<T> {query}
+   GetVectorUp()  :Vector3<T> {query}
+   Identity()  :Transform
+   Invert()  :bool
+   IsAffine()  :bool {query}
+   IsIdentity()  :bool {query}
+   operator-()  :Transform {query}
+   operator-(Transform&)  :Transform {query}
+   operator-=(Transform&)  :Transform&
+   operator!=(Transform&)  :bool {query}
+   operator()(U32, U32)  :T& {query}
+   operator()(U32, U32)  :T&
+   operator*(Transform&)  :Transform {query}
+   operator*(T)  :Transform {query}
+   operator*=(Transform&)  :Transform&
+   operator*=(T)  :Transform&
+   operator[](U32)  :T& {query}
+   operator[](U32)  :T&
+   operator+(Transform&)  :Transform {query}
+   operator+=(Transform&)  :Transform&
+   operator==(Transform&)  :bool {query}
+   Regenerate()  :void
+   Rotate(T, Vector3<T>&)  :void
+   Rotate(Transform&, Vector3<T>&)  :Vector3<T>
+   RotateX(T)  :void
+   RotateY(T)  :void
+   RotateZ(T)  :void
+   Scale(T, T, T)  :void
+   Scale(Vector3<T>&)  :void
+   Scale(T)  :void
+   ScaleX(T)  :void
+   ScaleY(T)  :void
+   ScaleZ(T)  :void
+   SetIdentity()  :void
+   SetRotation(T, T, T)  :void
+   SetRotation(Vector3<T>&)  :void
+   SetRotation(T, Vector3<T>&)  :void
+   SetRotation(Vector3<T>&, Vector3<T>&, Vector3<T>&)  :void
+   SetRotationX(T)  :void
+   SetRotationY(T)  :void
+   SetRotationZ(T)  :void
+   SetScale(T)  :void
+   SetScale(T, T, T)  :void
+   SetScale(Vector3<T>&)  :void
+   SetTranslation(T, T, T)  :void
+   SetTranslation(Vector3<T>&)  :void
+   Transform()
+   Transform(T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T)
+   Transform(T*)
+   Transform(Transform&, Vector3<T>&)  :Vector3<T>
+   Translate(T, T, T)  :void
+   Translate(Vector3<T>&)  :void
+   TranslateX(T)  :void
+   TranslateY(T)  :void
+   TranslateZ(T)  :void
+   Transpose()  :void
```

```
T : typename
Vector2
+   AxisX()  :Vector2
+   AxisY()  :Vector2
+   Cross(Vector2&, Vector2&)  :T
+   Dot(Vector2&, Vector2&)  :T
+   GetLength()  :T {query}
+   GetLengthSquared()  :T {query}
+   IsZero()  :bool {query}
+   Max(Vector2&, Vector2&)  :Vector2
+   Min(Vector2&, Vector2&)  :Vector2
+   MinMax(Vector2&, Vector2&, Vector2*, U32)  :void
+   Normalize()  :void
+   operator-()  :Vector2 {query}
+   operator-(Vector2&)  :Vector2 {query}
+   operator-(T)  :Vector2 {query}
+   operator-=(Vector2&)  :Vector2&
+   operator-=(T)  :Vector2&
+   operator!=(Vector2&)  :bool {query}
+   operator!=(T)  :bool {query}
+   operator*(Vector2&)  :Vector2 {query}
+   operator*(T)  :Vector2 {query}
+   operator*=(Vector2&)  :Vector2&
+   operator*=(T)  :Vector2&
+   operator/(Vector2&)  :Vector2 {query}
+   operator/(T)  :Vector2 {query}
+   operator/=(Vector2&)  :Vector2&
+   operator/=(T)  :Vector2&
+   operator[](size_t)  :T& {query}
+   operator[](size_t)  :T&
+   operator^(Vector2&)  :T {query}
+   operator+(Vector2&)  :Vector2 {query}
+   operator+(T)  :Vector2 {query}
+   operator+=(Vector2&)  :Vector2&
+   operator+=(T)  :Vector2&
+   operator==(Vector2&)  :bool {query}
+   operator==(T)  :bool {query}
+   Set(T)  :void
+   Set(T, T)  :void
+   SetZero()  :void
+   Sign()  :void
+   Vector2()
+   Vector2(T)
+   Vector2(T, T)
+   ZeroVector()  :Vector2
```

```
T : typename
Vector3
+   AxisX()  :Vector3
+   AxisY()  :Vector3
+   AxisZ()  :Vector3
+   Cross(Vector3&, Vector3&)  :Vector3
+   Dot(Vector3&, Vector3&)  :T
+   GetLength()  :T {query}
+   GetLengthSquared()  :T {query}
+   IsZero()  :bool {query}
+   Max(Vector3&, Vector3&)  :Vector3
+   Min(Vector3&, Vector3&)  :Vector3
+   MinMax(Vector3&, Vector3&, Vector3<T>*, U32)  :void
+   Normalize()  :void
+   operator-()  :Vector3 {query}
+   operator-(Vector3&)  :Vector3 {query}
+   operator-(T)  :Vector3 {query}
+   operator-=(Vector3&)  :Vector3&
+   operator-=(T)  :Vector3&
+   operator!=(Vector3&)  :bool {query}
+   operator!=(T)  :bool {query}
+   operator*(Vector3&)  :Vector3 {query}
+   operator*(T)  :Vector3 {query}
+   operator*=(Vector3&)  :Vector3&
+   operator*=(T)  :Vector3&
+   operator/(Vector3&)  :Vector3 {query}
+   operator/(T)  :Vector3 {query}
+   operator/=(Vector3&)  :Vector3&
+   operator/=(T)  :Vector3&
+   operator[](size_t)  :T& {query}
+   operator[](size_t)  :T&
+   operator^(Vector3&)  :Vector3 {query}
+   operator+(Vector3&)  :Vector3 {query}
+   operator+(T)  :Vector3 {query}
+   operator+=(Vector3&)  :Vector3&
+   operator+=(T)  :Vector3&
+   operator==(Vector3&)  :bool {query}
+   operator==(T)  :bool {query}
+   Set(T)  :void
+   Set(T, T, T)  :void
+   SetZero()  :void
+   Sign()  :void
+   Vector3()
+   Vector3(T)
+   Vector3(T, T, T)
+   ZeroVector()  :Vector3
```

```
T : typename
Vector4
+   AxisW()  :Vector4
+   AxisX()  :Vector4
+   AxisY()  :Vector4
+   AxisZ()  :Vector4
+   Dot(Vector4&, Vector4&)  :T
+   GetLength()  :T {query}
+   GetLengthSquared()  :T {query}
+   IsZero()  :bool {query}
+   Max(Vector4&, Vector4&)  :Vector4
+   Min(Vector4&, Vector4&)  :Vector4
+   MinMax(Vector4&, Vector4&, Vector4<T>*, U32)  :void
+   Normalize()  :void
+   operator-()  :Vector4 {query}
+   operator-(Vector4&)  :Vector4 {query}
+   operator-(T)  :Vector4 {query}
+   operator-=(Vector4&)  :Vector4&
+   operator-=(T)  :Vector4&
+   operator!=(Vector4&)  :bool {query}
+   operator!=(T)  :bool {query}
+   operator*(Vector4&)  :Vector4 {query}
+   operator*(T)  :Vector4 {query}
+   operator*=(Vector4&)  :Vector4&
+   operator*=(T)  :Vector4&
+   operator/(Vector4&)  :Vector4 {query}
+   operator/(T)  :Vector4 {query}
+   operator/=(Vector4&)  :Vector4&
+   operator/=(T)  :Vector4&
+   operator[](size_t)  :T& {query}
+   operator[](size_t)  :T&
+   operator+(Vector4&)  :Vector4 {query}
+   operator+(T)  :Vector4 {query}
+   operator+=(Vector4&)  :Vector4&
+   operator+=(T)  :Vector4&
+   operator==(Vector4&)  :bool {query}
+   operator==(T)  :bool {query}
+   Set(T)  :void
+   Set(T, T, T, T)  :void
+   SetZero()  :void
+   Sign()  :void
+   Vector4()
+   Vector4(T)
+   Vector4(T, T, T, T)
+   ZeroVector()  :Vector4
```