Skip to content

Freestyle Utilities (freestyle.utils)

This module contains helper functions used for Freestyle style module writing.

SUBMODULES

freestyle.utils submodule (freestyle.utils.ContextFunctions)

freestyle.utils.**getCurrentScene()**

Returns the current scene.

RETURNS:

The current scene.

RETURN TYPE:

`bpy.types.Scene`

freestyle.utils.**integrate(func, it, it_end, integration_type)**

Returns a single value from a set of values evaluated at each 0D element of this 1D element.

PARAMETERS:

- **func** (`UnaryFunction0D`) – The UnaryFunction0D used to compute a value at each Interface0D.
- **it** (`Interface0DIterator`) – The Interface0DIterator used to iterate over the 0D elements of this 1D element. The integration will occur over the 0D elements starting from the one pointed by it.
- **it_end** (`Interface0DIterator`) – The Interface0DIterator pointing the end of the 0D elements of the 1D element.
- **integration_type** (`IntegrationType`) – The integration method used to compute a single value from a set of values.

RETURNS:

The single value obtained for the 1D element. The return value type is float if func is of the `UnaryFunction0DDouble` or `UnaryFunction0DFloat` type, and int if func is of the `UnaryFunction0DUnsigned` type.

RETURN TYPE:

int | float

freestyle.utils.**angle_x_normal(it: Interface0DIterator)**

unsigned angle between a Point’s normal and the X axis, in radians

freestyle.utils.**bound(lower, x, higher)**

Returns x bounded by a maximum and minimum value. Equivalent to:

return min(max(x, lower), higher)

freestyle.utils.**bounding_box(stroke)**

Returns the maximum and minimum coordinates (the bounding box) of the stroke’s vertices

freestyle.utils.**curvature_from_stroke_vertex(svert)**

The 3D curvature of an stroke vertex’ underlying geometry

The result is None or in the range [-inf, inf]

freestyle.utils.**find_matching_vertex(id, it)**

Finds the matching vertex, or returns None.

freestyle.utils.**get_chain_length(ve, orientation)**

Returns the 2d length of a given ViewEdge.

freestyle.utils.**get_object_name(stroke)**

Returns the name of the object that this stroke is drawn on.

freestyle.utils.**get_strokes()**

Get all strokes that are currently available

freestyle.utils.**get_test_stroke()**

Returns a static stroke object for testing

freestyle.utils.**is_poly_clockwise(stroke)**

True if the stroke is orientated in a clockwise way, False otherwise

freestyle.utils.**iter_distance_along_stroke(stroke)**

Yields the absolute distance along the stroke up to the current vertex.

freestyle.utils.**iter_distance_from_camera(stroke, range_min, range_max, normfac)**

Yields the distance to the camera relative to the maximum possible distance for every stroke vertex, constrained by given minimum and maximum values.

freestyle.utils.**iter_distance_from_object(stroke, location, range_min, range_max, normfac)**

yields the distance to the given object relative to the maximum possible distance for every stroke vertex, constrained by given minimum and maximu values.

freestyle.utils.**iter_material_value(stroke, func, attribute)**

Yields a specific material attribute from the vertex' underlying material.

freestyle.utils.**iter_t2d_along_stroke(stroke)**

Yields the progress along the stroke.

freestyle.utils.**material_from_fedge(fe)**

get the diffuse RGBA color from an FEdge

freestyle.utils.**normal_at_I0D(it: Interface0DIterator) → Vector**

Normal at an Interface0D object. In contrast to Normal2DF0D this

function uses the actual data instead of underlying Fedge objects.

freestyle.utils.**pairwise(iterable, types={<class 'Stroke'>, <class 'StrokeVertexIterator'>})**

Yields a tuple containing the previous and current object

freestyle.utils.**rgb_to_bw(r, g, b)**

Method to convert rgb to a bw intensity value.

freestyle.utils.**simplify(points, tolerance)**

Simplifies a set of points

freestyle.utils.**stroke_curvature(it)**

Compute the 2D curvature at the stroke vertex pointed by the iterator 'it'. K = 1 / R where R is the radius of the circle going through the current vertex and its neighbors

freestyle.utils.**stroke_normal(stroke)**

Compute the 2D normal at the stroke vertex pointed by the iterator 'it'. It is noted that Normal2DF0D computes normals based on underlying FEdges instead, which is inappropriate for strokes when they have already been modified by stroke geometry modifiers.

The returned normals are dynamic: they update when the vertex position (and therefore the vertex normal) changes. for use in geometry modifiers it advised to cast this generator function to a tuple or list

freestyle.utils.**tripplewise(iterable)**

Yields a tuple containing the current object and its immediate neighbors

class freestyle.utils.**BoundingBox**

Object representing a bounding box consisting out of 2 2D vectors

inside(other)

True if self inside other, False otherwise

class freestyle.utils.**StrokeCollector**

Collects and Stores stroke objects

shade(stroke)