UF_FACET_add_facet_to_model (view source)

Defined in: uf_facet.h

Overview

Adds a facet to a faceted model. The "adjacent_facet_ids" array contains a facet id for each edge in the new facet. A value of UF_FACET_NULL_FACET_ID means that that edge of the new facet is not to be made adjacent to any existing facet. Note that the adjacent facets must contain an edge corresponding to that in the new facet, and if they do not then an error is returned and no facet is created. This function can return the error code UF_FACET_err_zero_facets_produced when given a polygon with adjacent duplicate vertices.

Return

Return code:
0 = No error
not 0 = Error code, which includes
UF_FACET_err_non_manifold
UF_FACET_err_bad_adjacency
UF_FACET_err_facet_not_planar
UF_FACET_err_zero_facets_produced

Environment

Internal and External

See Also

UF_FACET_del_facet_from_model
UF_FACET_set_adjacent_facet

For an example please refer to example

Required License(s)

assemblies

```
int UF_FACET_add_facet_to_model
(
    tag_t model,
    int num_vertices,
    double vertices [ ] [ 3 ] ,
    double normals [ ] [ 3 ] ,
    int adjacent_facet_ids [ ] ,
    int * new_facet_id
)
```

tag_t	**model**	Input	The model to which the facet is to be added
int	**num_vertices**	Input	The number of vertices in the new facet
double	**vertices [ ] [3]**	Input	The vertices of the new facet. Each element of three double in this argument represents a 3-space (X, Y, Z) position.
double	**normals [ ] [3]**	Input	The vertex normals for the new facet. Each element of three double in this argument represents a 3-space (X, Y, Z) normal vector If this is specified as NULL then the facet normal will be computed and used for all of the vertices in the new facet.

int	**adjacent_facet_ids [ ]**	Input	The facets which are adjacent to the new facet at each of its edges. If there is no adjacent facet to the edge then UF_FACET_NULL_FACET_ID should be specified for that edge.
int *	**new_facet_id**	Output	The id of the new facet created

---

UF_FACET_ask_adjacent_facet (view source)

Defined in: uf_facet.h

Overview

Enquires the facet adjacent to a specified edge of a specified facet. If there is no facet adjacent to the facet edge specified then UF_FACET_NULL_FACET_ID is returned in adjacent_facet_id and the contents of edge_id_in_adjacent_facet is undefined.

Environment

Internal and External

See Also

UF_FACET_add_facet_to_model
UF_FACET_set_adjacent_facet
For an example please refer to example

Required License(s)

gateway

```
int UF_FACET_ask_adjacent_facet
(
    tag_t model,
    int facet_id,
    int edge_id,
    int * adjacent_facet_id,
    int * edge_id_in_adjacent_facet
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int	**edge_id**	Input	The edge of which the adjacent facet is being inquired.
int *	**adjacent_facet_id**	Output	The adjacent facet
int *	**edge_id_in_adjacent_facet**	Output	The edge in the adjacent facet which is adjacent to edge_id in facet_id.

UF_FACET_ask_available_solid (view source)

Defined in: uf_facet.h

Overview
Enquire the solid model on which a faceted model is based. If the faceted model is not based on any solid or if the solid is excluded by reference set or is in an unloaded component then NULL_TAG is returned in "solid"

Environment
Internal and External

Required License(s)
gateway

```
int UF_FACET_ask_available_solid
(
    tag_t model,
    tag_t * solid
)
```

tag_t	**model**	Input	The model.
tag_t *	**solid**	Output	IF "model has an available defining solid then the tag of that solid, otherwise NULL_TAG".

UF_FACET_ask_default_parameters (view source)

Defined in: uf_facet.h

Overview
Enquires the default faceting parameters for creating faceted models from solid bodies and faces.

Environment
Internal and External

See Also
UF_FACET_set_default_parameters
UF_FACET_ask_model_parameters

For example please refer to example

Required License(s)
gateway

```
int UF_FACET_ask_default_parameters
(
    UF_FACET_parameters_p_t parameters
)
```

UF_FACET_parameters_p_t	**parameters**	Output	Default parameters for faceting solids.

UF_FACET_ask_edge_convexity (view source)

Defined in: uf_facet.h

Overview

Enquires the convexity of an edge between two facets. The convexity values are UF_FACET_IS_CONVEX, UF_IS_FACET_CONCAVE, and UF_FACET_CONVEXITY_NOT_DETERMINED; the last of these indicate that the two facets adjacent to the edge are very nearly coplanar and therefore whether the edge should be considered convex or concave is unclear. (Note that an ifail is returned if the specified edge is open.)

Environment

Internal and External

See Also

For an example please refer to example

Required License(s)

gateway

```
int UF_FACET_ask_edge_convexity
(
    tag_t model,
    int facet_id,
    int edge_in_facet,
    int * convexity
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int	**edge_in_facet**	Input	The edge for which convexity is being inquired.
int *	**convexity**	Output	The convexity of the edge.

UF_FACET_ask_face_id_of_facet (view source)

Defined in: uf_facet.h

Overview

Enquire the face id of a facet. Output is a pointer to UF_FACET_NULL_FACE_ID if the facet is not associated with any face in the model. If the facet_id is out of range for this model an error will occur.

Environment

Internal and External

History

Originally released in V16.0

Required License(s)

gateway

int UF_FACET_ask_face_id_of_facet
(
tag_t model,
int facet_id,
int * face_id
)

tag_t	**model**	Input	The model.
int	**facet_id**	Input	The id of the facet.
int *	**face_id**	Output	The id of the face.

UF_FACET_ask_face_id_of_solid_face (view source)

Defined in: uf_facet.h

Overview

Enquire the face id of a face, given its tag. Output is a pointer to UF_FACET_NULL_FACE_ID if the face is not present in the model or if face tags have not been stored (note that face tags are stored only if "store_face_tags" is set to "true" when faceting a solid).

Environment

Internal and External

History

Originally released in V16.0

Required License(s)

gateway

int UF_FACET_ask_face_id_of_solid_face
(
tag_t model,
tag_t face_tag,
int * face_id
)

tag_t	**model**	Input	The model.
tag_t	**face_tag**	Input	The solid face tag.
int *	**face_id**	Output	The face id.

UF_FACET_ask_max_facet_verts (view source)

Defined in: uf_facet.h

Overview
Enquires the maximum number of vertices of any facet in the model.
This routine can be used to ensure that the arrays required by UF_FACET_ask_vertices_of_facet and UF_FACET_ask_normals_of_facet are sufficiently large to contain the result.

Environment
Internal and External

See Also
For an example please refer to example

Required License(s)
gateway

```
int UF_FACET_ask_max_facet_verts
(
    tag_t model,
    int * num_facets
)
```

tag_t	**model**	Input	The model.
int *	**num_facets**	Output	The maximum number of facets of any facet in the model.

UF_FACET_ask_model_parameters (view source)

Defined in: uf_facet.h

Overview
Determines faceting parameters when given a faceted model which was generated from (and remains associated with a solid).

Environment
Internal and External

See Also
UF_FACET_facet_solid
UF_FACET_update_model
For an example please refer to example

Required License(s)
gateway

```
int UF_FACET_ask_model_parameters
(
    tag_t model,
    UF_FACET_parameters_p_t parameters
)
```

tag_t	**model**	Input	The model.

UF_FACET_parameters_p_t	**parameters**	Output	The parameters to facet the solid associated with "model".

UF_FACET_ask_models_of_solid (view source)

Defined in: uf_facet.h

Overview
Enquire the faceted model(s) of a specified solid body or face.

Environment
Internal and External

See Also
UF_FACET_ask_solid_of_model
For an example please refer to example

Required License(s)
gateway

```
int UF_FACET_ask_models_of_solid
(
    tag_t solid,
    int * n_faceted_models,
    tag_t * * faceted_models
)
```

tag_t	**solid**	Input	A solid body or face
int *	**n_faceted_models**	Output	The number of faceted models associated with "solid".
tag_t * *	**faceted_models**	Output to UF_*free*	The tags of the faceted models associated with "solid". This space is allocated by the called routine and must be freed by the caller using UF_free.

UF_FACET_ask_n_facets_in_model (view source)

Defined in: uf_facet.h

Overview
Enquires the number of facets in a model.

Environment
Internal and External

See Also
For an example please refer to example

Required License(s)
gateway

```
int UF_FACET_ask_n_facets_in_model
(
    tag_t model,
    int * num_facets
)
```

tag_t	**model**	Input	The model.
int *	**num_facets**	Output	The number of facets in the model.

UF_FACET_ask_normals_of_facet (view source)

Defined in: uf_facet.h

Overview

Enquires the vertex normals of the facet with the specified facet id. The caller is responsible for making sure that the normals array is large enough to contain all of the normal data.

Environment

Internal and External

See Also

UF_FACET_ask_vertices_of_facet
UF_FACET_ask_num_verts_in_facet
UF_FACET_ask_max_facet_verts
For an example please refer to example

Required License(s)

gateway

```
int UF_FACET_ask_normals_of_facet
(
    tag_t model,
    int facet_id,
    int * num_vertices,
    double normals [ ] [ 3 ]
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int *	**num_vertices**	Output	The number of vertices in the facet.

double	**normals [ ] [3]**	Output	The vertex normals for this facet. Each element of three double in this argument represents a 3-space (X, Y, Z) normal vector. Note that this is passed by the caller, not allocated by the routine, and the caller is responsible for ensuring that this array is sufficiently long, which can be done by previously calling UF_FACET_ask_num_verts_in_facet or by calling UF_FACET_ask_max_facet_verts.

UF_FACET_ask_num_faces (view source)

Defined in: uf_facet.h

Overview
Enquire the number of faces in a model. Typically only useful if the model has been derived from a solid model. For models constructed directly, the output will be zero.

Environment
Internal and External

History
Originally released in V16.0

Required License(s)
gateway

```
int UF_FACET_ask_num_faces
(
    tag_t model,
    int * num_faces
)
```

tag_t	**model**	Input	The model.
int *	**num_faces**	Output	The number of faces in the model.

UF_FACET_ask_num_facets_in_face (view source)

Defined in: uf_facet.h

Overview
Enquire the number of facets in a face. If the face_id is out of range for this model an error will occur.

Environment
Internal and External

History
Originally released in V16.0

Required License(s)
gateway

int UF_FACET_ask_num_facets_in_face
(
tag_t model,
int face_id,
int * num_facets
)

tag_t	**model**	Input	The model.
int	**face_id**	Input	The id of the face.
int *	**num_facets**	Output	The number of facets in the face.

UF_FACET_ask_num_verts_in_facet (view source)

Defined in: uf_facet.h

Overview
Enquires the number of vertices in a facet.

Environment
Internal and External

See Also
For an example please refer to example

Required License(s)
gateway

int UF_FACET_ask_num_verts_in_facet
(
tag_t model,
int facet_id,
int * num_vertices
)

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int *	**num_vertices**	Output	The number of vertices in the facet.

UF_FACET_ask_params_of_facet (view source)

Defined in: uf_facet.h

Overview
Enquires the vertex parameters of the facet with the specified facet id.
The caller is responsible for making sure that the params array
is large enough to contain all of the parameter data.

Environment
Internal and External

See Also
UF_FACET_ask_vertices_of_facet
UF_FACET_ask_num_verts_in_facet
UF_FACET_ask_max_facet_verts

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_FACET_ask_params_of_facet
(
    tag_t model,
    int facet_id,
    int * num_params,
    double params [ ] [ 3 ]
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int *	**num_params**	Output	The number of vertices in the facet.
double	**params [ ] [3]**	Output	The vertex parameters for this facet. Each element of three double in this argument represents the parametric values (u,v,t) at a vertex. Note that this is passed by the caller, not allocated by the routine, and the caller is responsible for ensuring that this array is sufficiently long, which can be done by previously calling UF_FACET_ask_num_verts_in_facet or by calling UF_FACET_ask_max_facet_verts.

UF_FACET_ask_plane_equation (view source)

Defined in: uf_facet.h

Overview
Enquire the plane equation of a facet.

Environment
Internal and External

Required License(s)
gateway

```
int UF_FACET_ask_plane_equation
(
    tag_t model,
    int facet_id,
    double plane_normal [ 3 ] ,
    double * d_coefficient
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
double	**plane_normal [3]**	Output	The plane normal for the facet. This is a 3-space (X, Y, Z) normal vector. The (X, Y, Z) of this vector provide the A, B, and C coefficients respectively of the plane equation: Ax + By + Cz +D = 0
double *	**d_coefficient**	Output	The d_coefficient for the plane of the facet. This provides the D coefficient in the above equation.

UF_FACET_ask_solid_face_of_face_id (view source)

Defined in: uf_facet.h

Overview

Enquire the solid face tag of a face, given its id. If the face id is out of range for this model, an error will occur. If face tags have not been stored and the face id is in range, then NULL_TAG is output. (Note that face tags are stored only if "store_face_tags" is set to "true" when faceting a solid.)

Environment

Internal and External

History

Originally released in V16.0

Required License(s)

gateway

```
int UF_FACET_ask_solid_face_of_face_id
(
    tag_t model,
    int face_id,
    tag_t * face_tag
)
```

tag_t	**model**	Input	The model.
int	**face_id**	Input	The id of the face.
tag_t *	**face_tag**	Output	The solid face tag.

UF_FACET_ask_solid_face_of_facet (view source)

Defined in: uf_facet.h

Overview
Enquire the solid face tag of a facet. Output is a pointer to NULL_TAG if the facet is not associated with any face in the model. If the facet_id is out of range for this model an error will occur.

Note that the model must have been faceted with "store_face_tags" set to "true" for this function to return a meaningful result (otherwise NULL_TAG will be returned for all facets).

Environment
Internal and External

History
Originally released in V16.0

Required License(s)
gateway

```
int UF_FACET_ask_solid_face_of_facet
(
    tag_t model,
    int facet_id,
    tag_t * face_tag
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
tag_t *	**face_tag**	Output	The solid face tag.

UF_FACET_ask_solid_of_model (view source)

Defined in: uf_facet.h

Overview
Enquires the solid model on which a faceted model is based. If the faceted model is not based on any solid then NULL_TAG is returned.

Environment
Internal and External

See Also
UF_FACET_ask_models_of_solid
For an example please refer to example

Required License(s)
gateway

```
int UF_FACET_ask_solid_of_model
(
    tag_t model,
    tag_t * solid
)
```

tag_t	**model**	Input	The faceted model.
tag_t *	**solid**	Output	The associated solid. This is NULL_TAG if there is no associated model.

UF_FACET_ask_surface_data_for_face (view source)

Defined in: uf_facet.h

Overview
Enquire the surface data of a face. If the face_id is out of range for this model an error will occur.

Environment
Internal and External

History
Originally released in NX8.5.1

Required License(s)
gateway

```
int UF_FACET_ask_surface_data_for_face
(
    tag_t facet_face,
    int * type,
    double pos [ 3 ] ,
    double dir [ 3 ] ,
    double * radius,
    double * radius_data,
    logical * sense,
    logical * from_cached_analytics
)
```

tag_t	**facet_face**	Input	The tag of the facet face
int *	**type**	Output	The surface type. The value of this type corresponds with the type in the NxOpen::Face::FaceType enum. If not analytic (plane/cylinder/cone/torus/sphere) then this will be FaceTypeUndefined.
double	**pos [3]**	Output	The origin: 3D point: Point on plane Point on axis(Cylinder, Cone) Centre point(Sphere, Torus)
double	**dir [3]**	Output	The normal direction for planes and axis for others. NA for Sphere.

double *	**radius**	Output	Radius: Not valid for planes Radius(Cylinder, Sphere, Torus(Major Axis), Cone(Radius in plane passing through position)
double *	**radius_data**	Output	Not valid for planes, cylinders, spheres for cones this is the half angle, for torus it is the minor radius.
logical *	**sense**	Output	The sense of the face with respect to the normal direction
logical *	**from_cached_analytics**	Output	true - If the surface data was obtained from the lightweight analytic data cached in the facet model and false - if it was derived using an inferencing algorithm.

UF_FACET_ask_vertex_convexity (view source)

Defined in: uf_facet.h

Overview

Enquires the convexity of a vertex in a facet. The convexity values are UF_FACET_IS_CONVEX, UF_FACET_IS_CONCAVE, and UF_FACET_CONVEXITY_NOT_DETERMINED; the last of these indicates that the two edges from the vertex are very nearly colinear and therefore whether the vertex should be considered convex or concave is unclear.

Environment

Internal and External

See Also

UF_FACET_ask_edge_convexity
For an example please refer to example

Required License(s)

gateway

```
int UF_FACET_ask_vertex_convexity
(
    tag_t model,
    int facet_id,
    int vertex_in_facet,
    int * convexity
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int	**vertex_in_facet**	Input	The vertex for which convexity is being inquired.
int *	**convexity**	Output	The convexity of the vertex

UF_FACET_ask_vertices_of_facet (view source)

Defined in: uf_facet.h

Overview

Enquires the vertices of the facet with the specified facet id.
The caller is responsible for assuring that the vertices array is large enought to hold all of the vertices of the facet.

Environment

Internal and External

See Also

UF_FACET_ask_normals_of_facet
UF_FACET_ask_max_facet_verts
UF_FACET_ask_num_verts_in_facet
For an example please refer to example

Required License(s)

gateway

```
int UF_FACET_ask_vertices_of_facet
(
    tag_t model,
    int facet_id,
    int * num_vertices,
    double vertices [ ] [ 3 ]
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
int *	**num_vertices**	Output	The number of vertices in the facet.
double	**vertices [ ] [3]**	Output	The vertices for this facet. Each element of three double in this argument represents a 3-space (X, Y, Z) position. Note that this is passed by the caller, not allocated by the routine, and the caller is responsible for ensuring that this array is sufficiently long, which can be done by previously calling UF_FACET_ask_num_verts_in_facet or by calling UF_FACET_ask_max_facet_verts.

UF_FACET_create_model (view source)

Defined in: uf_facet.h

Overview

Creates an empty faceted model which is not associated with any solid.

Environment
Internal and External

See Also
UF_FACET_add_facet_to_model
UF_FACET_model_edits_done

For an example please refer to example

Required License(s)
assemblies

int UF_FACET_create_model
(
tag_t object_in_part,
tag_t * model
)

tag_t	**object_in_part**	Input	An object in the part in which the faceted model is to be created.
tag_t *	**model**	Output	The new faceted model.

UF_FACET_cycle_facets (view source)

Defined in: uf_facet.h

Overview
Iterates over the facets in a faceted model. The cycle is started using the code UF_FACET_NULL_FACET_ID and this is also returned when there are no further facets in the model. Note that although "facet_id" is an integer, there is no guarantee that "facet_id+1" is a valid facet id.

Environment
Internal and External

See Also
For an example please refer to example

Required License(s)
assemblies

int UF_FACET_cycle_facets
(
tag_t model,
int * facet_id
)

tag_t	**model**	Input	The faceted model to be cycled.
int *	**facet_id**	Input / Output	On input the id of the current facet in the model. Pass in UF_FACET_NULL_FACET_ID to start the cycle.

			On output the id of the next facet in the model, if there are no more facets, UF_FACET_NULL_FACET_ID is returned.

UF_FACET_cycle_facets_in_face (view source)

Defined in: uf_facet.h

Overview

Iterates over the facets in a face. The cycle is started using the code UF_FACET_NULL_FACET_ID and this is also returned when there are no further facets in the face. Note that although "facet_id" is an integer, there is no guarantee that "facet_id+1" is a valid facet id.
If the face_id is outside the range for this model, an error occurs.

Environment

Internal and External

History

Originally released in V16.0

Required License(s)

assemblies

```
int UF_FACET_cycle_facets_in_face
(
    tag_t model,
    int face_id,
    int * facet_id
)
```

tag_t	**model**	Input	The model.
int	**face_id**	Input	The face whose facets are to be cycled.
int *	**facet_id**	Input / Output	On input the id of the current facet in the model. Pass in UF_FACET_NULL_FACET_ID to start the cycle. On output the id of the next facet in the model, if there are no more facets, UF_FACET_NULL_FACET_ID is returned.

UF_FACET_del_facet_from_model (view source)

Defined in: uf_facet.h

Overview

Deletes a facet from a model. This operation leaves a hole in the faceted model where the facet was deleted, unless the deleted facet was at the edge of a sheet. If the model previously enclosed space then it will no longer do so. If required at a future time, the hole can be patched either by using UF_FACET_add_facet_to_model to add new facets to the model, or by using UF_FACET_set_adjacent_facet to

stitch existing facets onto the model.

Environment
Internal and External

See Also
UF_FACET_add_facet_to_model

For an example please refer to example

Required License(s)
assemblies

int UF_FACET_del_facet_from_model
(
tag_t model,
int facet_id
)

tag_t	**model**	Input	The model from which the facet is to be deleted.
int	**facet_id**	Input	The facet id of the facet to be deleted

---

UF_FACET_delete_all_facets_from_model (view source)

Defined in: uf_facet.h

Overview
Deletes all of the facets from a model.

Environment
Internal and External

See Also
UF_FACET_add_facet_to_model

Required License(s)
assemblies

int UF_FACET_delete_all_facets_from_model
(
tag_t model
)

tag_t	**model**	Input	The model from which the facet is to be deleted.

UF_FACET_disassoc_from_solid (view source)

Defined in: uf_facet.h

Overview
Disassociates a faceted model from its generating solid so that it stands on its own. The link is established only if the faceted model was created using UF_FACET_facet_solid.

Environment
Internal and External

See Also
UF_FACET_facet_solid

For an example please refer to example

Required License(s)
assemblies

int UF_FACET_disassoc_from_solid
(
tag_t model
)

tag_t	**model**	Input	The model which is to be disassociated from its solid definition.

UF_FACET_facet_solid (view source)

Defined in: uf_facet.h

Overview
Produces a faceted model from a solid body or face.
NOTE: The faceted body created is added to the reference set of the work occurrence if used in context.

Return
Return code:
0 = No error
not 0 = Error code, which includes
UF_FACET_err_non_manifold

Environment
Internal and External

See Also
UF_FACET_ask_default_parameters
UF_FACET_ask_model_parameters
UF_FACET_is_model_up_to_date
UF_FACET_update_model

For example please refer to example

Required License(s)
assemblies

int UF_FACET_facet_solid
(

```
    tag_t solid_entity,
    UF_FACET_parameters_p_t parameters,
    tag_t * facet_model
)
```

tag_t	**solid_entity**	Input	The solid to be faceted
UF_FACET_parameters_p_t	**parameters**	Input	The parameters to be used to facet the solid. If NULL is specified then the defaults are used.
tag_t *	**facet_model**	Output	The faceted model produced

UF_FACET_find_edge_in_facet (view source)

Defined in: uf_facet.h

Overview
Determines if a specified edge appears in a specified facet.

Return
Return code:
0 = No error
not 0 = Error code which includes
UF_FACET_err_edge_not_found

Environment
Internal and External

See Also
UF_FACET_ask_adjacent_facet
For an example please refer to example

Required License(s)
assemblies

```
int UF_FACET_find_edge_in_facet
(
    tag_t model,
    int facet_id,
    double vertex_1 [ 3 ] ,
    double vertex_2 [ 3 ] ,
    int * sense,
    int * edge_id
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
double	**vertex_1 [3]**	Input	The first vertex in the edge. This is a 3-space (X, Y, Z) position.
double	**vertex_2 [3]**	Input	The second vertex in the edge. This is a 3-space (X, Y, Z) position.

int *	**sense**	Output	The sense of the edge in the facet. This is: 0 if the edge appears as vertex_1 -> vertex_2 1 if the edge appears as vertex_2 -> vertex_1
int *	**edge_id**	Output	The id of the edge in the facet.

UF_FACET_is_facet_convex (view source)

Defined in: uf_facet.h

Overview
Enquires if a facet is convex. A facet is convex if none of its vertices are concave.

Environment
Internal and External

See Also
For an example please refer to example

Required License(s)
gateway

```
int UF_FACET_is_facet_convex
(
    tag_t model,
    int facet_id,
    logical * is_convex
)
```

tag_t	**model**	Input	The model
int	**facet_id**	Input	The id of the facet.
logical *	**is_convex**	Output	TRUE = facet has no concave vertices FALSE = facet has concave vertices

---

UF_FACET_is_model_convex (view source)

Defined in: uf_facet.h

Overview
Enquires if a faceted model is convex. A model is convex if none of its edges are concave.

Environment
Internal and External

See Also
For an example please refer to example

Required License(s)
gateway

int UF_FACET_is_model_convex
(
tag_t model,
logical * is_convex
)

tag_t	**model**	Input	The model
logical *	**is_convex**	Output	TRUE if the model has no concave edges and FALSE otherwise.

UF_FACET_is_model_up_to_date (view source)

Defined in: uf_facet.h

Overview
Enquires if a faceted model is up to date.

Environment
Internal and External

See Also
UF_FACET_update_model
For an example please refer to example

Required License(s)
assemblies

int UF_FACET_is_model_up_to_date
(
tag_t model,
logical * up_to_date
)

tag_t	**model**	Input	The faceted model.
logical *	**up_to_date**	Output	Whether or not "model" is up_to_date.

UF_FACET_model_edits_done (view source)

Defined in: uf_facet.h

Overview
Indicates that edits to the specified faceted model are now complete. This allows the system to delete the temporary data structures which it creates to facilitate editing. This function should be called after the program has completed a sequence of editing operations (other than

creation and update) on a faceted model.

Note that if this routine is not called then no harm results, other than that the program will consume more memory than is necessary.

Environment
Internal and External

See Also
UF_FACET_create_model

For an example please refer to example

Required License(s)
assemblies

int UF_FACET_model_edits_done
(
tag_t model
)

tag_t	**model**	Input	The model which we have finished editing.

UF_FACET_rebuild_adjacencies (view source)

Defined in: uf_facet.h

Overview
Determines the adjacencies between the facets of a faceted body. The routine uses the individual facet definitions and determines which have shared edges. It is particularly useful where the faceted body has been constructed using UF_FACET_add_facet_to_model without specifying the adjacent facets, and may prove convenient when writing code to import faceted models from elsewhere.

Environment
Internal and External

Required License(s)
assemblies

int UF_FACET_rebuild_adjacencies
(
tag_t model
)

tag_t	**model**	Input	The tag of the faceted model

UF_FACET_set_adjacent_facet (view source)

Defined in: uf_facet.h

Overview
Sets a facet as being adjacent to another facet along a specified edge.

Return
Return code:
0 = No error
not 0 = Error code, which includes
UF_FACET_err_bad_adjacency

Environment
Internal and External

See Also
UF_FACET_add_facet_to_model

For an example please refer to example

Required License(s)
assemblies

```
int UF_FACET_set_adjacent_facet
(
    tag_t model,
    int facet_id,
    int edge,
    int adjacent_facet_id
)
```

tag_t	**model**	Input	The model in which the two facets reside
int	**facet_id**	Input	The facet for which the adjacent facet is to be specified
int	**edge**	Input	The edge in facet_id for which and adjacent facet is to be specified.
int	**adjacent_facet_id**	Input	The adjacent facet.

UF_FACET_set_default_parameters (view source)

Defined in: uf_facet.h

Overview
Sets the default faceting parameters for creating faceted models from solid bodies and faces.

Environment
Internal and External

See Also
UF_FACET_ask_default_parameters
UF_FACET_ask_model_parameters

Required License(s)

assemblies

int UF_FACET_set_default_parameters
(
UF_FACET_parameters_p_t parameters
)

UF_FACET_parameters_p_t	**parameters**	Input	Default parameters for faceting solids.

UF_FACET_set_vertex_of_facet (view source)

Defined in: uf_facet.h

Overview

Change the location of a vertex of a facet in a faceted body. Note that the normals at this vertex are not changed.

Environment

Internal and External

Required License(s)

assemblies

int UF_FACET_set_vertex_of_facet
(
tag_t model,
int facet_id,
int vertex_in_facet,
double location [3]
)

tag_t	**model**	Input	The model.
int	**facet_id**	Input	The id of the facet.
int	**vertex_in_facet**	Input	The vertex whose location is being changed.
double	**location [3]**	Input	The new location of the vertex.

UF_FACET_update_model (view source)

Defined in: uf_facet.h

Overview

Updates a faceted model if it was generated from a solid. This function returns an error if the faceted model was not generated from a solid. If NULL is specified for the faceting parameters then those used to facet the solid previously are used.

If the model is already up to date with the same faceting parameters,

then this function performs no action. If the model is not derived from a solid then an error is returned.

Return

Return code:
0 = No error
not 0 = Error code, which includes
UF_FACET_err_non_manifold

Environment

Internal and External

See Also

UF_FACET_ask_default_parameters
UF_FACET_ask_model_parameters
UF_FACET_is_model_up_to_date
For an example please refer to example

History

In V14.0, the parameters argument data type was changed to UF_FACET_parameters_p_t.

Required License(s)

assemblies

int UF_FACET_update_model
(
tag_t model,
UF_FACET_parameters_p_t parameters
)

tag_t	**model**	Input	The faceted model to update.
UF_FACET_parameters_p_t	**parameters**	Input	The parameters to be used for faceting the model. If NULL is specified then the original parameters are used.