UF_SO_ask_3_scalars_of_point (view source)

Defined in: uf_so.h

Overview
Find the x, y and z values of a smart point.

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_ask_3_scalars_of_point
(
    tag_t so_point,
    tag_t scalars [ 3 ]
)
```

tag_t	**so_point**	Input	Tag of the smart point
tag_t	**scalars [3]**	Output	Array that holds the three scalars

---

UF_SO_ask_assy_ctxt_part_occ (view source)

Defined in: uf_so.h

Overview
Given a to_part_occ for the assy_context xform, get the corresponding from_part_occ (i.e. one that is part of the same assembly part occ tree). The to_part_occ must be one whose prototype part contains the assy_context xform. Returns an error if given a non-assy_context xform object.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_SO_ask_assy_ctxt_part_occ
(
    const tag_t assy_context_xform,
    const tag_t to_part_occ,
    tag_p_t from_part_occ
)
```

const tag_t	**assy_context_xform**	Input	Tag of assy_context transform

const tag_t	**to_part_occ**	Input	to_part_occ giving assy context.
tag_p_t	**from_part_occ**	Output	from_part_occ corresponding to given to_part_occ.

UF_SO_ask_children (view source)

Defined in: uf_so.h

Overview
Asks smart object children of object.

Environment
Internal and External

History
Original Release was in V15.0.

Required License(s)
gateway

```
int UF_SO_ask_children
(
    const tag_t object,
    int options,
    int * n_children,
    tag_p_t * children
)
```

const tag_t	**object**	Input	Tag of object on which to inquire smart object children.
int	**options**	Input	What kind of children (bit mask) UF_SO_ASK_SO_CHILDREN - Ask referencing smart objects. UF_SO_ASK_CHILDREN_RECURSIVELY - Ask referencing objects recursively. UF_SO_ASK_FEATURE_CHILDREN - Ask referencing features. UF_SO_ASK_ALL_CHILDREN - Ask all referencing features.
int *	**n_children**	Output	number of childen
tag_p_t *	**children**	Output to UF_*free*	Smart object children of object. This array must be freed by calling UF_free.

UF_SO_ask_direction_of_axis (view source)

Defined in: uf_so.h

Overview

Returns direction of axis.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_direction_of_axis
(
    const tag_t axis,
    double direction [ 3 ]
)
```

const tag_t	**axis**	Input	Tag of axis
double	**direction [3]**	Output	Value of axis direction

UF_SO_ask_direction_of_dirr (view source)

Defined in: uf_so.h

Overview
Returns the value of direction.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_direction_of_dirr
(
    const tag_t direction,
    double dir [ 3 ]
)
```

const tag_t	**direction**	Input	Tag of direction
double	**dir [3]**	Output	Value of direction

UF_SO_ask_dirr_on_surf (view source)

Defined in: uf_so.h

Overview
Inquire a direction on surface smart object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_ask_dirr_on_surf
(
    tag_t direction,
    UF_SO_dirr_on_surf_data_p_t dirr_on_surf_data
)
```

tag_t	**direction**	Input	object id of the direction on surface
UF_SO_dirr_on_surf_data_p_t	**dirr_on_surf_data**	Output	pointer to Open API data structure for direction on surface

---

UF_SO_ask_display_marker_of_point (view source)

Defined in: uf_so.h

Overview
Returns the display marker type for a point

Environment
Internal and External

History
Original Release was in V16.0.

Required License(s)
gateway

```
int UF_SO_ask_display_marker_of_point
(
    tag_t point,
    UF_DISP_poly_marker_t * disp_marker
)
```

tag_t	**point**	Input	Tag of the smart point
UF_DISP_poly_marker_t *	**disp_marker**	Output	Pointer to display marker type for the point

---

UF_SO_ask_double_of_scalar (view source)

Defined in: uf_so.h

Overview
Returns value of a scalar.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_double_of_scalar
(
    const tag_t scalar,
    double * dbl
)
```

const tag_t	**scalar**	Input	Tag of scalar
double *	**dbl**	Output	Value of scalar

UF_SO_ask_exp_of_scalar (view source)

Defined in: uf_so.h

Overview
Returns the expression of the smart scalar object which was created with UF_SO_create_scalar_exp.

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_ask_exp_of_scalar
(
    const tag_t scalar,
    tag_p_t exp
)
```

const tag_t	**scalar**	Input	Tag of scalar
tag_p_t	**exp**	Output	Expression of scalar

UF_SO_ask_matrix_of_xform (view source)

Defined in: uf_so.h

Overview
Returns matrix of transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_matrix_of_xform
(
    tag_t xform,
    double matrix [ 16 ]
)
```

tag_t	**xform**	Input	Tag of transform
double	**matrix [16]**	Output	4 x 4 transform matrix

UF_SO_ask_offset_curve_cvtr (view source)

Defined in: uf_so.h

Overview
Inquire a curvature on curve smart offset object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_ask_offset_curve_cvtr
(
    tag_t curvature,
    UF_SO_offset_curve_cvtr_data_p_t offset_curve_cvtr_data
)
```

tag_t	**curvature**	Input	object id of the smart offset for curvature on curve

UF_SO_offset_curve_cvtr_data_p_t	**offset_curve_cvtr_data**	Output	pointer to Open API data structure for curvature on curve

---

UF_SO_ask_offset_of_offset (view source)

Defined in: uf_so.h

Overview
Returns offset vector of specified offset object.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_offset_of_offset
(
    const tag_t offset,
    double offset_vec [ 3 ]
)
```

const tag_t	**offset**	Input	Tag of offset
double	**offset_vec [3]**	Output	Value of offset

UF_SO_ask_offset_surf_cvtr (view source)

Defined in: uf_so.h

Overview
Inquire a curvature on surface smart offset object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_ask_offset_surf_cvtr
(
    tag_t curvature,
    UF_SO_offset_surf_cvtr_data_p_t offset_surf_cvtr_data
```

)

tag_t	**curvature**	Input	object id of the smart offset for curvature on surface
UF_SO_offset_surf_cvtr_data_p_t	**offset_surf_cvtr_data**	Output	pointer to Open API data structure for curvature on surface

---

UF_SO_ask_parent_status (view source)

Defined in: uf_so.h

Overview

Returns the parent status for the specified smart object.
See parent status bit masks

Environment

Internal and External

History

Original Release was in V13.0.

Required License(s)

gateway

```
int UF_SO_ask_parent_status
(
    const tag_t so,
    int * parent_status
)
```

const tag_t	**so**	Input	Tag of smart object
int *	**parent_status**	Output	Status of parent

UF_SO_ask_parents (view source)

Defined in: uf_so.h

Overview

Asks parents of smart object. Note that the following abbreviations or terms are used in the descriptions.

lwos = Light weight ocurrences. These are not referenced but are selectable and are displayable.

hwos = Heavy weight ocurrences. These are referenced and have data.

stub = References an object in another part file that is not loaded.

Environment
Internal and External

History
Original Release was in V15.0.

Required License(s)
gateway

```
int UF_SO_ask_parents
(
    const tag_t so,
    int options,
    int * n_parents,
    tag_p_t * parents
)
```

const tag_t	**so**	Input	Tag of smart object on which to inquire parents
int	**options**	Input	What kind of parents (bit mask) UF_SO_ASK_EXP_PARENTS - Ask refernced expressions. UF_SO_ASK_SO_PARENTS - Ask referenced smart objects. UF_SO_ASK_WIREFRAME_PARENTS - Ask referenced wire frame geometry (hwos). UF_SO_ASK_ALL_PARENTS - Ask referenced objects including lwos and stubs. UF_SO_ASK_PARENTS_RECURSIVELY - Ask referenced objects recursively. UF_SO_ASK_UNLOADED_PARENTS - Return unloaded parents.
int *	**n_parents**	Output	Number of parents
tag_p_t *	**parents**	Output to UF_*free*	Parents of smart object

---

UF_SO_ask_point_of_axis (view source)

Defined in: uf_so.h

Overview
Returns point of axis.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_point_of_axis
(
    const tag_t axis,
    double point [ 3 ]
)
```

const tag_t	**axis**	Input	Tag of axis
double	**point [3]**	Output	Value of axis point

UF_SO_ask_point_of_xform (view source)

Defined in: uf_so.h

Overview
Returns point of transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_point_of_xform
(
    const tag_t xform,
    double point [ 3 ]
)
```

const tag_t	**xform**	Input	Tag of transform
double	**point [3]**	Output	Value of transform point

UF_SO_ask_scale_of_xform (view source)

Defined in: uf_so.h

Overview
Returns scale of transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_scale_of_xform
(
    tag_t xform,
    double * scale
)
```

tag_t	**xform**	Input	Tag of transform
double *	**scale**	Output	Value of transform scale

UF_SO_ask_spline (view source)

Defined in: uf_so.h

Overview
Inquire a general smart spline object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_ask_spline
(
    tag_t spline,
    UF_SO_spline_data_p_t spline_data
)
```

tag_t	**spline**	Input	object id of the smart spline
UF_SO_spline_data_p_t	**spline_data**	Output	pointer to Open API data structure for general spline

---

UF_SO_ask_update_error_code (view source)

Defined in: uf_so.h

Overview
Returns the smart object update error code.

Environment
Internal and External

History

Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_ask_update_error_code
(
tag_t so,
int * update_error_code
)

tag_t	**so**	Input	Tag of smart object
int *	**update_error_code**	Output	Status of parent

UF_SO_ask_visibility_option (view source)

Defined in: uf_so.h

Overview
Returns the visibility option for the specified smart object.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_ask_visibility_option
(
const tag_t so,
UF_SO_visibility_option_p_t visibility_option
)

const tag_t	**so**	Input	Tag of smart object
UF_SO_visibility_option_p_t	**visibility_option**	Output	Status of visibility option

---

UF_SO_ask_x_direction_of_xform (view source)

Defined in: uf_so.h

Overview
Returns x-direction of transform.

Environment

Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_ask_x_direction_of_xform
(
tag_t xform,
double x_direction [3]
)

tag_t	**xform**	Input	Tag of transform
double	**x_direction [3]**	Output	Value of transform X direction

---

UF_SO_ask_y_direction_of_xform (view source)

Defined in: uf_so.h

Overview
Returns y-direction of transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_ask_y_direction_of_xform
(
tag_t xform,
double y_direction [3]
)

tag_t	**xform**	Input	Tag of transform
double	**y_direction [3]**	Output	Value of transform Y direction

---

UF_SO_ask_z_direction_of_xform (view source)

Defined in: uf_so.h

Overview

Returns z-direction of transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_ask_z_direction_of_xform
(
    tag_t xform,
    double z_direction [ 3 ]
)
```

tag_t	**xform**	Input	Tag of transform
double	**z_direction [3]**	Output	Value of transform Z direction

---

UF_SO_create_arc_center_2_pnts (view source)

Defined in: uf_so.h

Overview
Creates a smart arc via center and two points.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_arc_center_2_pnts
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t points [ 3 ] ,
    tag_p_t arc
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**points [3]**	Input	Array of tags of three points (center, start, and end point)
tag_p_t	**arc**	Output	Pointer to tag of smart arc

UF_SO_create_arc_radius_angles (view source)

Defined in: uf_so.h

Overview

Creates a smart arc via transformation, radius, and angles. To obtain the transformation tag use any of the UF_SO_create_xform_ routines. The radius and angles are scalars; use any of the UF_SO_create_scalar_ routines to get tags for these scalars.

Environment

Internal and External

History

Original Release was in V13.0.

Required License(s)

gateway

```
int UF_SO_create_arc_radius_angles
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t xform,
    const tag_t radius,
    const tag_t angles [ 2 ] ,
    tag_p_t arc
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**xform**	Input	Tag of transformation
const tag_t	**radius**	Input	Tag of radius
const tag_t	**angles [2]**	Input	Tags of start and end angles
tag_p_t	**arc**	Output	Pointer to tag of smart arc

UF_SO_create_arc_three_points (view source)

Defined in: uf_so.h

Overview

Creates a smart arc through three points.

Environment

Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_arc_three_points
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t points [ 3 ] ,
    tag_p_t arc
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**points [3]**	Input	Array of tags of three points
tag_p_t	**arc**	Output	Pointer to tag of smart arc

UF_SO_create_arc_xform_2_points (view source)

Defined in: uf_so.h

Overview
Creates a smart arc via two points and a transform. The arc always lies on the xy plane of the xform. The two points are projected onto the xy plane to give start and end points for the arc.

Environment
Internal and External

History
Original Release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_arc_xform_2_points
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t xform,
    const tag_t points [ 2 ] ,
    tag_p_t arc
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option

const tag_t	**xform**	Input	Tag of transform
const tag_t	**points [2]**	Input	Array of tags of two points (start and end point)
tag_p_t	**arc**	Output	Pointer to tag of smart arc

UF_SO_create_axis_doubles (view source)

Defined in: uf_so.h

Overview
Creates a double axis.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_axis_doubles
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double point [ 3 ] ,
    const double direction [ 3 ] ,
    tag_p_t axis
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const double	**point [3]**	Input	Axis point
const double	**direction [3]**	Input	Axis direction
tag_p_t	**axis**	Output	Pointer to tag of axis

UF_SO_create_axis_extract (view source)

Defined in: uf_so.h

Overview
Creates a smart axis via extract axis with optional transformation.

Environment

Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_axis_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t axis,
    const tag_t xform,
    tag_p_t axis2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**axis**	Input	Tag of extract axis
const tag_t	**xform**	Input	Tag of transformation
tag_p_t	**axis2**	Output	Pointer to tag of smart axis

UF_SO_create_axis_point_dir (view source)

Defined in: uf_so.h

Overview
Creates a point/direction axis.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_axis_point_dir
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t point,
    const tag_t direction,
    tag_p_t axis
)
```

const tag_t	**object_in_part**	Input	Tag of object in part

const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**point**	Input	Tag of axis point
const tag_t	**direction**	Input	Tag of axis direction
tag_p_t	**axis**	Output	Pointer to tag of smart axis

UF_SO_create_bcurve_thru_points (view source)

Defined in: uf_so.h

Overview
Create a smart b-spline curve through points

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_create_bcurve_thru_points
(
    UF_SO_update_option_t update_option,
    int num_of_points,
    tag_p_t points,
    double * point_parameters,
    int degree,
    int periodic,
    tag_t start_slope,
    tag_t end_slope,
    int start_slope_type,
    int end_slope_type,
    tag_p_t bcurve
)
```

UF_SO_update_option_t	**update_option**	Input	update option
int	**num_of_points**	Input	Number of points used to define the B-spline
tag_p_t	**points**	Input	Array of points
double *	**point_parameters**	Input	Array of user-specified parameters for points. The array length should be num_of_points for non-periodic curve and num_of_points+1 for periodic curves. The parameters need to start from 0.0 and be monotonically increasing; however, they need not to be normalized. Pass in NULL to use the default chord-length parametrization.
int	**degree**	Input	Degree of the B-spline curve

int	**periodic**	Input	Curve type - 0 = non-periodic and 1 = Periodic.
tag_t	**start_slope**	Input	Tag of a point whose coordinates are used as the start slope of a non-periodic cubic bcurve.
tag_t	**end_slope**	Input	Tag of a point whose coordinates are used as the end slope of a non-periodic cubic bcurve.
int	**start_slope_type**	Input	Start slope type of a non-periodic cubic bcurve. The available slope types are: -1 = no slope 0 = auto slope 1 = use only direction of slope 2 = use both direction and magnitude of slope
int	**end_slope_type**	Input	End slope type of a non-periodic cubic bcurve. The available slope types are the same as above.
tag_p_t	**bcurve**	Output	The smart B-spline

UF_SO_create_curve_extract (view source)

Defined in: uf_so.h

Overview
Creates a smart curve via extract curve with optional transformation.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_curve_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t curve1,
    const int type,
    const int subtype,
    const tag_t xform,
    tag_p_t curve2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**curve1**	Input	Tag of curve from which to extract

const int	**type**	Input	Curve type
const int	**subtype**	Input	Curve subtype
const tag_t	**xform**	Input	Tag of transformation
tag_p_t	**curve2**	Output	Pointer to tag of smart curve

UF_SO_create_dirr_axis_of_conic (view source)

Defined in: uf_so.h

Overview
Creates direction via axis of conic.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_axis_of_conic
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t conic,
    const logical flip,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**conic**	Input	Tag of plane (curve/edge)
const logical	**flip**	Input	TRUE = Flip direction FALSE = Do not flip direction
tag_p_t	**direction**	Output	Pointer to tag of smart direction

---

UF_SO_create_dirr_doubles (view source)

Defined in: uf_so.h

Overview

Creates a double direction.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_doubles
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double direction [ 3 ] ,
    tag_p_t dirr
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const double	**direction [3]**	Input	Real direction vector
tag_p_t	**dirr**	Output	Pointer to tag of smart direction

UF_SO_create_dirr_doubles_pnt (view source)

Defined in: uf_so.h

Overview
Create a double direction. The user defined real point of the double direction is used for display.

Environment
Internal and External

History
Original Release was in V16.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_doubles_pnt
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double point [ 3 ] ,
    const double direction [ 3 ] ,
    tag_p_t dirr
)
```

const tag_t	**object_in_part**	Input	Object in the part to create the double direction in.
const UF_SO_update_option_t	**update_option**	Input	update option
const double	**point [3]**	Input	real point of direction
const double	**direction [3]**	Input	real direction vector
tag_p_t	**dirr**	Output	Double direction object that was created.

UF_SO_create_dirr_extract (view source)

Defined in: uf_so.h

Overview
Creates direction via extract direction with optional transformation.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t dir,
    const tag_t xform,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**dir**	Input	Tag of direction
const tag_t	**xform**	Input	Tag of transformation (NULL_TAG => identity)
tag_p_t	**direction**	Output	Pointer to tag of smart direction

---

UF_SO_create_dirr_line (view source)

Defined in: uf_so.h

Overview
Creates direction of line.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_line
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t line,
    const logical flip,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**line**	Input	Tag of line (curve/edge/axis)
const logical	**flip**	Input	TRUE = Flip direction FALSE = Do not flip direction
tag_p_t	**direction**	Output	Pointer to tag of smart direction

---

UF_SO_create_dirr_normal_to_surface_point (view source)

Defined in: uf_so.h

Overview
Creates a direction normal to a surface at a given point on that surface.

Environment
Internal and External

History
Original Release was in V16.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_normal_to_surface_point
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
```

```
    const tag_t face,
    const tag_t point,
    const logical flip,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**face**	Input	Tag of the face.
const tag_t	**point**	Input	Tag of the point on the face where the normal is to be created.
const logical	**flip**	Input	TRUE = Flip direction FALSE = Do not flip direction
tag_p_t	**direction**	Output	Pointer to the tag of a smart direction.

UF_SO_create_dirr_on_curve (view source)

Defined in: uf_so.h

Overview

Creates a direction on a curve using t value.

Note: This function can not be used to compute the normal for a line or a spline made from two points. This is due to the fact that lines have no associated csys so there is no unique normal.

Environment

Internal and External

History

Original Release was in V13.0.

Required License(s)

gateway

```
int UF_SO_create_dirr_on_curve
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t curve,
    const tag_t t,
    const UF_SO_dirr_on_curve_option_t option,
    const logical flip,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part

const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**curve**	Input	Tag of curve
const tag_t	**t**	Input	Tag of parameter object
const UF_SO_dirr_on_curve_option_t	**option**	Input	Tangent/Normal/Binormal
const logical	**flip**	Input	TRUE = Flip direction FALSE = Do not flip direction
tag_p_t	**direction**	Output	Pointer to tag of direction

UF_SO_create_dirr_on_surf (view source)

Defined in: uf_so.h

Overview
Create a direction on surface smart object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_on_surf
(
    tag_t object_in_part,
    UF_SO_update_option_t update_option,
    UF_SO_dirr_on_surf_data_p_t dirr_on_surf_data,
    tag_t * direction
)
```

tag_t	**object_in_part**	Input	the part context
UF_SO_update_option_t	**update_option**	Input	update option
UF_SO_dirr_on_surf_data_p_t	**dirr_on_surf_data**	Input	pointer to Open API data structure for direction on surface
tag_t *	**direction**	Output	pointer to object id of the direction on surface

---

UF_SO_create_dirr_plane (view source)

Defined in: uf_so.h

Overview
Creates direction of plane.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_plane
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t plane,
    const logical flip,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**plane**	Input	Tag of plane (face/datum)
const logical	**flip**	Input	TRUE = Flip direction FALSE = Do not flip direction
tag_p_t	**direction**	Output	Pointer to tag of smart direction

---

UF_SO_create_dirr_surface_axis (view source)

Defined in: uf_so.h

Overview
Creates direction via axis of surface.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_surface_axis
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
```

```
    const tag_t conic,
    const logical flip,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**conic**	Input	Tag of face
const logical	**flip**	Input	TRUE = Flip direction FALSE = Do not flip direction
tag_p_t	**direction**	Output	Pointer to tag of smart direction

---

UF_SO_create_dirr_two_dirs (view source)

Defined in: uf_so.h

Overview
Creates a direction via two directions. The output direction is the cross product of the two input directions.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_two_dirs
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t directions [ 2 ] ,
    tag_p_t direction
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**directions [2]**	Input	Array of tags of two directions
tag_p_t	**direction**	Output	Pointer to tag of smart direction

---

UF_SO_create_dirr_two_points (view source)

Defined in: uf_so.h

Overview
Creates a smart direction via two points.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_dirr_two_points
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t points [ 2 ] ,
    tag_p_t dirr
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**points [2]**	Input	Two points
tag_p_t	**dirr**	Output	Pointer to tag of smart direction

UF_SO_create_line_two_points (view source)

Defined in: uf_so.h

Overview
Creates a smart line between two points.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_line_two_points
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t points [ 2 ] ,
    tag_p_t line
```

```
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**points [2]**	Input	Array of tags of two points
tag_p_t	**line**	Output	Pointer to tag of smart line

UF_SO_create_offset_3_scalars (view source)

Defined in: uf_so.h

Overview
Creates a smart offset using three smart scalars.

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_create_offset_3_scalars
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    tag_t xyz [ 3 ] ,
    tag_p_t offset
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
tag_t	**xyz [3]**	Input	Array of tags of smart scalars
tag_p_t	**offset**	Output	Pointer to smart offset

UF_SO_create_offset_curve_cvtr (view source)

Defined in: uf_so.h

Overview
Create a curvature on curve smart offset object.

Environment
Internal and External

History

Originally released in V18.0.

Required License(s)
gateway

int UF_SO_create_offset_curve_cvtr
(
tag_t object_in_part,
UF_SO_update_option_t update_option,
UF_SO_offset_curve_cvtr_data_p_t offset_curve_cvtr_data,
tag_t * curvature
)

tag_t	**object_in_part**	Input	the part context
UF_SO_update_option_t	**update_option**	Input	update option
UF_SO_offset_curve_cvtr_data_p_t	**offset_curve_cvtr_data**	Input	pointer to Open API data structure for curvature on curve
tag_t *	**curvature**	Output	pointer to object id of the smart offset for curvature on curve

---

UF_SO_create_offset_cylindrical (view source)

Defined in: uf_so.h

Overview
Creates a smart cylindrical offset. The radius, angle, and zdelta are the three parameters of cylindrical coordinates.

Environment
Internal and External

Required License(s)
gateway

int UF_SO_create_offset_cylindrical
(
const tag_t object_in_part,
const UF_SO_update_option_t update_option,
tag_t radius,
tag_t angle,
tag_t zdelta,
tag_p_t offset
)

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
tag_t	**radius**	Input	Tag of scalar radius

tag_t	**angle**	Input	Tag of scalar angle in radians
tag_t	**zdelta**	Input	Tag of scalar delta along axis of cylinder.
tag_p_t	**offset**	Output	Pointer to tag of vector offset

UF_SO_create_offset_dir_dist (view source)

Defined in: uf_so.h

Overview
Creates a direction/distance offset.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_offset_dir_dist
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t direction,
    const tag_t distance,
    tag_p_t offset
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**direction**	Input	Tag of direction vector
const tag_t	**distance**	Input	Tag of scalar offset distance
tag_p_t	**offset**	Output	Pointer to tag of direction/distance offset

---

UF_SO_create_offset_double (view source)

Defined in: uf_so.h

Overview
Creates a double offset.

Environment

Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_offset_double
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double offset1 [ 3 ] ,
    tag_p_t offset2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const double	**offset1 [3]**	Input	Real vector
tag_p_t	**offset2**	Output	Pointer to tag of double offset

UF_SO_create_offset_double_pnt (view source)

Defined in: uf_so.h

Overview
Create a double offset. The user defined real point of the double offset is used for display.

Environment
Internal and External

History
Original Release was in V16.0.

Required License(s)
gateway

```
int UF_SO_create_offset_double_pnt
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double point [ 3 ] ,
    const double offset1 [ 3 ] ,
    tag_p_t offset2
)
```

const tag_t	**object_in_part**	Input	Object in the part where the double offset will be created.

const UF_SO_update_option_t	**update_option**	Input	update option
const double	**point [3]**	Input	real point of offset
const double	**offset1 [3]**	Input	real offset vector
tag_p_t	**offset2**	Output	Tag of the double offset created.

UF_SO_create_offset_extract (view source)

Defined in: uf_so.h

Overview
Creates a smart offset via extract offset with optional transformation.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_offset_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t offset1,
    const tag_t xform,
    tag_p_t offset2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**offset1**	Input	Tag of extract offset
const tag_t	**xform**	Input	Tag of transformation
tag_p_t	**offset2**	Output	Pointer to tag of smart offset

UF_SO_create_offset_spherical (view source)

Defined in: uf_so.h

Overview
Creates a smart spherical offset. The radius, angle1, and angle2 are the three parameters of spherical coordinates (r,q,f) respectively.

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_create_offset_spherical
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    tag_t radius,
    tag_t angle1,
    tag_t angle2,
    tag_p_t offset
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
tag_t	**radius**	Input	Tag of scalar spherical radius
tag_t	**angle1**	Input	Tag of scalar angle (longitude) in radians
tag_t	**angle2**	Input	Tag of scalar angle (colatitude) in radians
tag_p_t	**offset**	Output	Pointer to tag of vector offset

UF_SO_create_offset_surf_cvtr (view source)

Defined in: uf_so.h

Overview
Create a curvature on surface smart offset object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_create_offset_surf_cvtr
(
    tag_t object_in_part,
    UF_SO_update_option_t update_option,
    UF_SO_offset_surf_cvtr_data_p_t offset_surf_cvtr_data,
    tag_t * curvature
)
```

tag_t	**object_in_part**	Input	the part context
UF_SO_update_option_t	**update_option**	Input	update option
UF_SO_offset_surf_cvtr_data_p_t	**offset_surf_cvtr_data**	Input	pointer to Open API data structure for curvature on surface
tag_t *	**curvature**	Output	pointer to object id of the smart offset for curvature on surface

---

UF_SO_create_point_3_scalars (view source)

Defined in: uf_so.h

Overview
Creates a smart point via three scalars. The smart point update option must ensure that the parent scalars update before the point.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_point_3_scalars
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t xyz [ 3 ] ,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**xyz [3]**	Input	Array of tags of x,y, and z scalars
tag_p_t	**point**	Output	Pointer to tag of smart point

UF_SO_create_point_3_scalars_csys (view source)

Defined in: uf_so.h

Overview
Creates a smart point at an offset xyz from the given CSYS.

NOTE: If you use UF_SO_update_within_modeling as the update_option, then the input csys must also have its modeling option set to UF_SO_update_within_modeling.

Environment
Internal and External

History
The original release was in V18.0

Required License(s)
gateway

int UF_SO_create_point_3_scalars_csys
(
const tag_t object_in_part,
const tag_t csys_tag,
const tag_t xyz [3] ,
const UF_SO_update_option_t update_option,
tag_p_t point
)

const tag_t	**object_in_part**	Input	Tag of object in part
const tag_t	**csys_tag**	Input	Tag of the csys to reference the SO.
const tag_t	**xyz [3]**	Input	Array of tags of x, y and z scalars
const UF_SO_update_option_t	**update_option**	Input	Update option
tag_p_t	**point**	Output	Pointer to tag of smart point

UF_SO_create_point_along_curve (view source)

Defined in: uf_so.h

Overview
Creates a smart point along curve using a curve, point, and t (scalar offset along curve). This point is derived by finding the closest point on the given curve to the given point and then offsetting this point along the given curve by the arc length defined by t via absolute distance or relative percent.

Environment
Internal and External

Required License(s)
gateway

int UF_SO_create_point_along_curve
(
const tag_t object_in_part,
const UF_SO_update_option_t update_option,
const tag_t curve,
const tag_t point1,

```
    const tag_t t,
    const UF_SO_point_along_curve_option_t option,
    const logical flip,
    tag_p_t point2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**curve**	Input	Curve or edge
const tag_t	**point1**	Input	point
const tag_t	**t**	Input	Scalar offset along curve
const UF_SO_point_along_curve_option_t	**option**	Input	Absolute/Relative offset option. Can be one of the following enumerated constants: UF_SO_point_along_curve_distance UF_SO_point_along_curve_percent
const logical	**flip**	Input	If flip is set to TRUE, then t = -t
tag_p_t	**point2**	Output	Pointer to tag of smart point

UF_SO_create_point_conic_center (view source)

Defined in: uf_so.h

Overview
Creates a smart point via the center of a conic.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_point_conic_center
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t conic,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option

const tag_t	**conic**	Input	Tag of conic
tag_p_t	**point**	Output	Pointer to tag of point

UF_SO_create_point_extract (view source)

Defined in: uf_so.h

Overview
Creates a smart point via an extract point with optional transformation.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_point_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t point1,
    const tag_t xform,
    tag_p_t point2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**point1**	Input	Tag of point to be extracted
const tag_t	**xform**	Input	Tag of transformation
tag_p_t	**point2**	Output	Pointer to tag of smart point

UF_SO_create_point_extract_with_disp_marker (view source)

Defined in: uf_so.h

Overview
Creates a smart point via an extract point with a display marker and optional transformation.

Environment
Internal and External

History
Original Release was in V16.0.

Required License(s)
gateway

int UF_SO_create_point_extract_with_disp_marker
(
const tag_t object_in_part,
const UF_SO_update_option_t update_option,
const tag_t point1,
const tag_t xform,
UF_DISP_poly_marker_t disp_marker,
tag_p_t point2
)

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**point1**	Input	Tag of point to be extracted
const tag_t	**xform**	Input	Tag of transformation
UF_DISP_poly_marker_t	**disp_marker**	Input	Display marker type
tag_p_t	**point2**	Output	Pointer to tag of smart point

UF_SO_create_point_offset (view source)

Defined in: uf_so.h

Overview
Creates a smart point via a point and an offset.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_create_point_offset
(
const tag_t object_in_part,
const UF_SO_update_option_t update_option,
const tag_t base_point,
const tag_t offset,
tag_p_t point
)

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**base_point**	Input	Tag of base point
const tag_t	**offset**	Input	Tag of offset
tag_p_t	**point**	Output	Pointer to tag of point

UF_SO_create_point_on_arc_angle (view source)

Defined in: uf_so.h

Overview
Creates a smart point using an arc and angle.

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_create_point_on_arc_angle
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t arc,
    const tag_t angle,
    const tag_t xform,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**arc**	Input	Tag of arc
const tag_t	**angle**	Input	Tag of angle in radians
const tag_t	**xform**	Input	Tag of optional transform
tag_p_t	**point**	Output	Pointer to tag of smart point

UF_SO_create_point_on_axis (view source)

Defined in: uf_so.h

Overview

Creates a smart point on the axis of a surface by projecting a reference point onto the given axis. The axis tag should be that of an already defined axis smart object. The point tag argument may be that of a point smart object.

Environment
Internal and External

History
Original release was in v17.0

Required License(s)
gateway

```
int UF_SO_create_point_on_axis
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t ref_point,
    const tag_t axis,
    tag_p_t point_on_axis
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**ref_point**	Input	Tag of point to project onto axis
const tag_t	**axis**	Input	Tag of axis of cylindrical or conical face
tag_p_t	**point_on_axis**	Output	Pointer to tag of smart point

---

UF_SO_create_point_on_curve (view source)

Defined in: uf_so.h

Overview
Creates a smart point via a curve and scalar value t.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_point_on_curve
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t curve,
```

```
    const tag_t t,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**curve**	Input	Tag of curve
const tag_t	**t**	Input	Tag of scale
tag_p_t	**point**	Output	Pointer to tag of point

UF_SO_create_point_on_surface (view source)

Defined in: uf_so.h

Overview

Creates a smart point on a surface via uv scalars for the surface. Call one of the create scalar routines to obtain the input u and v parameters. For example, to create a point on the surface at (.25, .50) call UF_SO_create_scalar_double_dim() twice - once for the u value and once for the v value.

Note the u,v parameters are normalized from 0 to 1, so if you have read surface parameters with UF_MODL_ask_face_parms, or UF_MODL_ask_face_uv_minmax, you will have to normalize the parameters to the range of 0 to 1 prior to creating the smart point on the surface.

Environment

Internal and External

History

Original Release was in V13.0.

Required License(s)

gateway

```
int UF_SO_create_point_on_surface
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t face,
    const tag_t u,
    const tag_t v,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**face**	Input	Tag of face

const tag_t	**u**	Input	Tag of u scalar parameter
const tag_t	**v**	Input	Tag of v scalar parameter
tag_p_t	**point**	Output	Pointer to tag of smart point

UF_SO_create_point_surface_crv (view source)

Defined in: uf_so.h

Overview
Creates a smart point via surface/curve intersection.

Environment
Internal and External

Required License(s)
gateway

```
int UF_SO_create_point_surface_crv
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t face,
    const tag_t curve,
    const tag_t help_point1,
    const tag_t help_point2,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**face**	Input	Tag of surface
const tag_t	**curve**	Input	Tag of curve or edge
const tag_t	**help_point1**	Input	Tag of point (optional)
const tag_t	**help_point2**	Input	Tag of point (optional)
tag_p_t	**point**	Output	Pointer to tag of smart point

UF_SO_create_point_two_curves (view source)

Defined in: uf_so.h

Overview

Creates a smart point via two curve intersection.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_point_two_curves
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t curve1,
    const tag_t curve2,
    const tag_t help_point1,
    const tag_t help_point2,
    tag_p_t point
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**curve1**	Input	Tag of first curve
const tag_t	**curve2**	Input	Tag of second curve
const tag_t	**help_point1**	Input	Tag of point
const tag_t	**help_point2**	Input	Tag of point
tag_p_t	**point**	Output	Pointer to tag of smart point

UF_SO_create_scalar_dist_2_pnts (view source)

Defined in: uf_so.h

Overview
Creates a smart scalar via distance between two points.

Environment
Internal and External

History
Original Release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_dist_2_pnts
(
```

```
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t points [ 2 ] ,
    tag_p_t scalar
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**points [2]**	Input	Array of tags for two points for point, curve, edge, face, or body.
tag_p_t	**scalar**	Output	Pointer to tag of smart scalar

UF_SO_create_scalar_double (view source)

Defined in: uf_so.h

Overview
Creates a double scalar.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_double
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double dbl,
    tag_p_t scalar
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const double	**dbl**	Input	Real constant
tag_p_t	**scalar**	Output	Pointer to double scalar

UF_SO_create_scalar_double_dim (view source)

Defined in: uf_so.h

Overview
Creates a dimensioned double scalar. A dimensioned double scalar should be used when the scalar represents a value with units, i.e. inches or millimeters. The value of the dimension can then be used to correctly convert the scalar when the part is converted from one unit to another.

Environment
Internal and External

History
Original Release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_double_dim
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double dbl,
    const UF_SO_scalar_dim_option_t dim,
    tag_p_t scalar
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const double	**dbl**	Input	Real constant
const UF_SO_scalar_dim_option_t	**dim**	Input	Dimension of scalar
tag_p_t	**scalar**	Output	Pointer to double scalar

UF_SO_create_scalar_exp (view source)

Defined in: uf_so.h

Overview
Creates a smart scalar via an expression.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_exp
(
```

```
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t exp,
    tag_p_t scalar
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**exp**	Input	Tag of expression
tag_p_t	**scalar**	Output	Pointer to smart scalar

UF_SO_create_scalar_exp_dim (view source)

Defined in: uf_so.h

Overview

Creates a dimensioned smart scalar via an expression. A dimensioned smart scalar should be used when the scalar represents a value with units, i.e. inches or millimeters. The value of the dimension can then be used to correctly convert the scalar when the part is converted from one unit to another.

Environment

Internal and External

History

Original Release was in V14.0.

Required License(s)

gateway

```
int UF_SO_create_scalar_exp_dim
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t exp,
    const UF_SO_scalar_dim_option_t dim,
    tag_p_t scalar
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**exp**	Input	Tag of expression
const UF_SO_scalar_dim_option_t	**dim**	Input	Dimension of scalar
tag_p_t	**scalar**	Output	Pointer to smart scalar

UF_SO_create_scalar_extract (view source)

Defined in: uf_so.h

Overview
Creates smart scalar via extract scalar with optional scale.

Environment
Internal and External

History
Original Release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t scalar1,
    const tag_t scale,
    tag_p_t scalar2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**scalar1**	Input	Tag of extracted scalar
const tag_t	**scale**	Input	Tag of transform or scalar (NULL_TAG implies the identity matrix for the transform)
tag_p_t	**scalar2**	Output	Pointer to tag of smart scalar

UF_SO_create_scalar_extract_dim (view source)

Defined in: uf_so.h

Overview
Creates a dimensioned smart scalar via extract scalar with optional scale. A dimensioned smart scalar should be used when the scalar represents a value with units, i.e. inches or millimeters. The value of the dimension can then be used to correctly convert the scalar when the part is converted from one unit to another.

Environment
Internal and External

History
Original Release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_extract_dim
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t scalar1,
    const tag_t scale,
    const UF_SO_scalar_dim_option_t dim,
    tag_p_t scalar2
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**scalar1**	Input	Tag of extracted scalar
const tag_t	**scale**	Input	Tag of transform or scalar (NULL_TAG implies the identity matrix for the transform)
const UF_SO_scalar_dim_option_t	**dim**	Input	Dimension of scalar
tag_p_t	**scalar2**	Output	Pointer to tag of smart scalar

UF_SO_create_scalar_length_crv (view source)

Defined in: uf_so.h

Overview
Creates a smart scalar via the length of a curve or edge.

Environment
Internal and External

History
Original Release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_scalar_length_crv
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t curve,
    tag_p_t scalar
)
```

const tag_t	**object_in_part**	Input	Tag of object in part

const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**curve**	Input	Tag of curve or edge
tag_p_t	**scalar**	Output	Pointer to tag of smart scalar

UF_SO_create_spline (view source)

Defined in: uf_so.h

Overview
Create a general smart spline object.

Environment
Internal and External

History
Originally released in V18.0.

Required License(s)
gateway

```
int UF_SO_create_spline
(
    UF_SO_update_option_t update_option,
    UF_SO_spline_data_p_t spline_data,
    tag_t * spline
)
```

UF_SO_update_option_t	**update_option**	Input	update option
UF_SO_spline_data_p_t	**spline_data**	Input	pointer to Open API data structure for general spline
tag_t *	**spline**	Output	pointer to object id of the created smart spline

---

UF_SO_create_xform_assy_ctxt (view source)

Defined in: uf_so.h

Overview
Create an assy_context smart transform, encapsulating the transform to go from one part occurrence in an assembly tree to another. Either from_part_occ or to_part_occ can be NULL_TAG, indicating that the transform goes to or from the root part occ of the specified part. If they are both non-NULL, they must be in the same part.
If from_part_occ and to_part_occ are the same, including if they are both NULL_TAG, then no assy_context_xform is created, and NULL_TAG is returned in the relevant argument.
If to_part_occ is not NULL_TAG, its prototype part has to be the part

that contains object_in_part.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_SO_create_xform_assy_ctxt
(
    const tag_t object_in_part,
    const tag_t from_part_occ,
    const tag_t to_part_occ,
    tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const tag_t	**from_part_occ**	Input	Beginning position for transform
const tag_t	**to_part_occ**	Input	End position for transform
tag_p_t	**xform**	Output	Pointer to tag of transform

UF_SO_create_xform_doubles (view source)

Defined in: uf_so.h

Overview
Creates a dumb double transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_xform_doubles
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const double point [ 3 ] ,
    const double x_direction [ 3 ] ,
    const double y_direction [ 3 ] ,
    const double scale,
    tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Unused - set to zero
const double	**point [3]**	Input	Point
const double	**x_direction [3]**	Input	X-direction
const double	**y_direction [3]**	Input	Y-direction
const double	**scale**	Input	Scale value
tag_p_t	**xform**	Output	Pointer to tag of smart transform

UF_SO_create_xform_extract (view source)

Defined in: uf_so.h

Overview
Creates a smart transform via extract transform with optional transformation.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_xform_extract
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t xform1,
    const tag_t xform2,
    tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**xform1**	Input	Tag of transform to extract from
const tag_t	**xform2**	Input	Tag of transform
tag_p_t	**xform**	Output	Pointer to tag of smart transform

UF_SO_create_xform_offset_xform (view source)

Defined in: uf_so.h

Overview

Creates a smart transform given a reference csys, point0, point1, rotational_offsets (if any) and an optional scale. It will translate then rotate. The csys will be rotated about the X axis, the Y axis then about the Z axis. Angles should be specified in degrees.

Environment

Internal and External

History

Original Release was in NX3

Required License(s)

gateway

```
int UF_SO_create_xform_offset_xform
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t csys,
    const tag_t point0,
    const tag_t point1,
    const tag_t rot_scalar_tags [ 3 ] ,
    const tag_t scale,
    const tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	determines the context
const UF_SO_update_option_t	**update_option**	Input	update option
const tag_t	**csys**	Input	reference csys
const tag_t	**point0**	Input	from point
const tag_t	**point1**	Input	to point
const tag_t	**rot_scalar_tags [3]**	Input	Rotation offset expressions tags
const tag_t	**scale**	Input	scalar (NULL_TAG => IDENTITY)
const tag_p_t	**xform**	Output	Smart xform

UF_SO_create_xform_pnt_xy_dirs (view source)

Defined in: uf_so.h

Overview

Creates a smart transform given a point, x and y directions, and an optional scale.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_xform_pnt_xy_dirs
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t point,
    const tag_t x_direction,
    const tag_t y_direction,
    const tag_t scale,
    tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option - must be the same as the update option used to create the input point
const tag_t	**point**	Input	Tag of point
const tag_t	**x_direction**	Input	Tag of x-direction
const tag_t	**y_direction**	Input	Tag of y-direction
const tag_t	**scale**	Input	Tag of scale value
tag_p_t	**xform**	Output	Pointer to tag of smart transform

UF_SO_create_xform_pnt_xz_dirs (view source)

Defined in: uf_so.h

Overview
Creates a smart transform given a point, x and z directions, and an optional scale.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_xform_pnt_xz_dirs
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t point,
    const tag_t x_direction,
    const tag_t z_direction,
    const tag_t scale,
    tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**point**	Input	Tag of point
const tag_t	**x_direction**	Input	Tag of x-direction
const tag_t	**z_direction**	Input	Tag of z-direction
const tag_t	**scale**	Input	Tag of scale value
tag_p_t	**xform**	Output	Pointer to tag of smart transform

UF_SO_create_xform_pnt_yz_dirs (view source)

Defined in: uf_so.h

Overview
Creates a smart transform given a point, y and z directions, and an optional scale.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_create_xform_pnt_yz_dirs
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t point,
    const tag_t y_direction,
    const tag_t z_direction,
    const tag_t scale,
    tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	Tag of object in part
const UF_SO_update_option_t	**update_option**	Input	Update option
const tag_t	**point**	Input	Tag of point
const tag_t	**y_direction**	Input	Tag of x-direction
const tag_t	**z_direction**	Input	Tag of z-direction
const tag_t	**scale**	Input	Tag of scale value
tag_p_t	**xform**	Output	Pointer to tag of smart transform

UF_SO_create_xform_three_planes (view source)

Defined in: uf_so.h

Overview
Creates a smart transform given a three planes/faces and the scale.

Environment
Internal and External

History
Original Release was in NX301

Required License(s)
gateway

```
int UF_SO_create_xform_three_planes
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t plane0,
    const tag_t plane1,
    const tag_t plane2,
    const tag_t scale,
    const tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	determines the context
const UF_SO_update_option_t	**update_option**	Input	update option
const tag_t	**plane0**	Input	first plane
const tag_t	**plane1**	Input	second plane
const tag_t	**plane2**	Input	third plane
const tag_t	**scale**	Input	scalar (NULL_TAG => IDENTITY)

const tag_p_t	**xform**	Output	Smart xform

UF_SO_create_xform_three_points (view source)

Defined in: uf_so.h

Overview
Creates a smart transform given a three points and the scale. The three points used to construct the xform must NOT be collinear. In other words, one point can't be collinear with the other 2 points.

Environment
Internal and External

History
Original Release was in NX301

Required License(s)
gateway

```
int UF_SO_create_xform_three_points
(
    const tag_t object_in_part,
    const UF_SO_update_option_t update_option,
    const tag_t point0,
    const tag_t point1,
    const tag_t point2,
    const tag_t scale,
    const tag_p_t xform
)
```

const tag_t	**object_in_part**	Input	determines the context
const UF_SO_update_option_t	**update_option**	Input	update option
const tag_t	**point0**	Input	origin point
const tag_t	**point1**	Input	x-dir point
const tag_t	**point2**	Input	y-dir point NOTE: The y-dir specified as an input may be modified slightly if it doesn not result in a true orthogonal direction given the specified origin and x-dir.
const tag_t	**scale**	Input	scalar (NULL_TAG => IDENTITY)
const tag_p_t	**xform**	Output	Smart xform

UF_SO_delete_non_deletables (view source)

Defined in: uf_so.h

Overview
Delete SO's that are condemned and not sleepy/promoted/occurrenced/referenced.

Environment
Internal and External

Required License(s)
gateway

int UF_SO_delete_non_deletables
(
tag_t part
)

tag_t	**part**	Input	part to process

UF_SO_delete_parms (view source)

Defined in: uf_so.h

Overview
Delete parms of smart object. All unreferenced
smart object parents are put on the delete list.

Environment
Internal and External

Required License(s)
gateway

int UF_SO_delete_parms
(
tag_t so
)

tag_t	**so**	Input	smart object

UF_SO_display (view source)

Defined in: uf_so.h

Overview
Displays a smart object with the specified options.

Environment
Internal and External

History

Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_display
(
const tag_t so,
const int view_option,
const int color_option,
const int color
)

const tag_t	**so**	Input	Tag of smart object to display
const int	**view_option**	Input	view option: UF_DISP_ALL_VIEWS_BUT_DRAWING UF_DISP_VIEW_OF_LAST_CURSOR UF_DISP_ALL_ACTIVE_VIEWS UF_DISP_WORK_VIEW_ONLY
const int	**color_option**	Input	color option: UF_DISP_USE_SYSTEM_COLOR UF_DISP_USE_BACKGROUND_COLOR UF_DISP_USE_ORIGINAL_COLOR UF_DISP_USE_SPECIFIED_COLOR
const int	**color**	Input	color option - used only when color option is UF_DISP_USE_SPECIFIED_COLOR. See uf_obj.h for valid color assignments.

UF_SO_display_parents (view source)

Defined in: uf_so.h

Overview
Displays a smart object's parents with the specified options.

Environment
Internal and External

Required License(s)
gateway

int UF_SO_display_parents
(
const tag_t so,
const int view_option,
const int color_option,
const int color
)

const tag_t	**so**	Input	Tag of smart object to display

const int	**view_option**	Input	view option: UF_DISP_ALL_VIEWS_BUT_DRAWING UF_DISP_VIEW_OF_LAST_CURSOR UF_DISP_ALL_ACTIVE_VIEWS UF_DISP_WORK_VIEW_ONLY
const int	**color_option**	Input	color option: UF_DISP_USE_SYSTEM_COLOR UF_DISP_USE_BACKGROUND_COLOR UF_DISP_USE_ORIGINAL_COLOR UF_DISP_USE_SPECIFIED_COLOR
const int	**color**	Input	color option - used only when color option is UF_DISP_USE_SPECIFIED_COLOR. See uf_obj.h for valid color assignments.

UF_SO_has_become_dumb (view source)

Defined in: uf_so.h

Overview

Determines if the given object has become dumb. This means that the object is no longer associative. For example, a smart point "has become dumb" after one calls UF_SO_delete_parms() on it.

Environment

Internal and External

See Also

UF_SO_delete_parms

History

Originally released in V15.0.

Required License(s)

gateway

```
int UF_SO_has_become_dumb
(
    tag_t so,
    logical * has_become_dumb
)
```

tag_t	**so**	Input	smart object to check if has become dumb
logical *	**has_become_dumb**	Output	TRUE if has become dumb, else FALSE

UF_SO_is_assy_ctxt_xform (view source)

Defined in: uf_so.h

Overview

Asks if an transform is an assy_context smart transform. Returns an error if given a non-transform object.

Environment
Internal and External

History
Original release was in V14.0.

Required License(s)
gateway

```
int UF_SO_is_assy_ctxt_xform
(
    const tag_t xform,
    logical * is_assy_xform
)
```

const tag_t	**xform**	Input	Tag of transform.
logical *	**is_assy_xform**	Output	True if transform is an assy_context transform, false otherwise

---

UF_SO_is_out_of_date (view source)

Defined in: uf_so.h

Overview
Determines if the given object is out of date. This means that the object did not update when it needed to. For example, a smart point "is out of date" after it does not update because some of its parents are not loaded.

Environment
Internal and External

See Also
UF_SO_ask_parent_status

History
Originally released in V15.0.

Required License(s)
gateway

```
int UF_SO_is_out_of_date
(
    tag_t so,
    logical * is_out_of_date
)
```

tag_t	**so**	Input	smart object to check if is out of date

logical *	**is_out_of_date**	Output	TRUE if is out of date, else FALSE

UF_SO_is_so (view source)

Defined in: uf_so.h

Overview

Queries if an object is a smart object and is parameterized.

These objects are "smart" so they know how they were constructed and will associatively update.

Environment

Internal and External

History

Original Release was in V13.0.

Required License(s)

gateway

```
int UF_SO_is_so
(
    const tag_t candidate,
    logical * is_so
)
```

const tag_t	**candidate**	Input	Tag of object
logical *	**is_so**	Output	TRUE = candidate is a smart object. FALSE = candidate is not a smart object.

UF_SO_is_subclass (view source)

Defined in: uf_so.h

Overview

Queries if an object inherits from the smart object class.
The object may or may not have parameters associated with it, i.e. it may or may not be smart itself. For that, call UF_SO_is_so.

Example:

```
logical is_subclass = FALSE;

UF_SO_is_so_subclass ( tag, &is_subclass );

If ( is_subclass )
{
logical has_become_dumb = FALSE;

UF_SO_has_become_dumb ( tag, &has_become_dumb );
```

```
if ( has_become_dumb )
{
Printf ("Smart Object (tag %u) has lost its associativity\n" tag );
}
}
```

Environment
Internal and External

History
Original Release was in V15.0.

Required License(s)
gateway

int UF_SO_is_subclass
(
const tag_t candidate,
logical * is_so
)

const tag_t	**candidate**	Input	Tag of object
logical *	**is_so**	Output	TRUE if candidate inherits from class SO FALSE otherwise

---

UF_SO_replace_parms (view source)

Defined in: uf_so.h

Overview
Replaces the parms of the old smart object with the parms of the new smart object and then deletes the new smart object.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_replace_parms
(
tag_t old_so,
tag_t new_so
)

tag_t	**old_so**	Input	Tag of smart object whose parms are to be replaced.

tag_t	**new_so**	Input	Tag of smart object whose parms are to be used as replacement parms.

UF_SO_set_direction_of_axis (view source)

Defined in: uf_so.h

Overview
Sets direction of dumb axis.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_direction_of_axis
(
    tag_t axis,
    const double new_direction [ 3 ]
)
```

tag_t	**axis**	Input	Tag of dumb axis
const double	**new_direction [3]**	Input	Value of axis direction

---

UF_SO_set_direction_of_dirr (view source)

Defined in: uf_so.h

Overview
Sets value of dumb direction.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_direction_of_dirr
(
    tag_t direction,
    const double dir [ 3 ]
)
```

tag_t	**direction**	Input	Tag of dumb direction
const double	**dir [3]**	Input	Value of direction.

UF_SO_set_display_marker_of_point (view source)

Defined in: uf_so.h

Overview
Sets/changes the display marker type for points created using UF_SO_create_point_extract_with_disp_marker()

Environment
Internal and External

History
Original Release was in V16.0.

Required License(s)
gateway

int UF_SO_set_display_marker_of_point
(
tag_t point,
UF_DISP_poly_marker_t disp_marker
)

tag_t	**point**	Input	Tag of the smart point
UF_DISP_poly_marker_t	**disp_marker**	Input	Display marker type for the point

---

UF_SO_set_double_of_scalar (view source)

Defined in: uf_so.h

Overview
Sets value of a dumb scalar.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

int UF_SO_set_double_of_scalar
(

```
    tag_t scalar,
    const double dbl
)
```

tag_t	**scalar**	Input	Tag of dumb scalar
const double	**dbl**	Input	Value of scalar

UF_SO_set_offset_of_offset (view source)

Defined in: uf_so.h

Overview
Sets offset of dumb offset.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_offset_of_offset
(
    tag_t offset,
    const double new_offset [ 3 ]
)
```

tag_t	**offset**	Input	Tag of dumb offset
const double	**new_offset [3]**	Input	Value of offset

UF_SO_set_point_of_axis (view source)

Defined in: uf_so.h

Overview
Sets point of dumb axis.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_point_of_axis
(
    tag_t axis,
    const double new_point [ 3 ]
)
```

tag_t	**axis**	Input	Tag of dumb axis
const double	**new_point [3]**	Input	Value of axis point

UF_SO_set_point_of_xform (view source)

Defined in: uf_so.h

Overview
Sets point of dumb transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_point_of_xform
(
    tag_t xform,
    const double point [ 3 ]
)
```

tag_t	**xform**	Input	Tag of dumb transform
const double	**point [3]**	Input	Value of transform point

UF_SO_set_scale_of_xform (view source)

Defined in: uf_so.h

Overview
Sets scale of dumb transform.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_scale_of_xform
(
    tag_t xform,
    const double scale
)
```

tag_t	**xform**	Input	Tag of dumb transform
const double	**scale**	Input	Value of transform scale

UF_SO_set_visibility_option (view source)

Defined in: uf_so.h

Overview
Sets the visibility option for the specified smart object. It is the responsibility of the calling routine to update the display.

Environment
Internal and External

History
Original Release was in V13.0.

Required License(s)
gateway

```
int UF_SO_set_visibility_option
(
    tag_t so,
    UF_SO_visibility_option_t visibility_option
)
```

tag_t	**so**	Input	Tag of smart object
UF_SO_visibility_option_t	**visibility_option**	Input	visibility option

---

UF_SO_set_xy_direction_of_xform (view source)

Defined in: uf_so.h

Overview
Sets x and y directions of dumb transform.

Environment
Internal and External

History

Original Release was in V13.0.

Required License(s)

gateway

```
int UF_SO_set_xy_direction_of_xform
(
    tag_t xform,
    const double x_direction [ 3 ] ,
    const double y_direction [ 3 ]
)
```

tag_t	**xform**	Input	Tag of dumb transform
const double	**x_direction [3]**	Input	Value of transform X direction
const double	**y_direction [3]**	Input	Value of transform Y direction

---