Appendix B.
Syntax Summary

Topic: Ignore

This appendix provides a syntax summary of reference items defined in the Component Name. There is a separate section for each item type (Scheme extension, test harness command, etc.), and the items within each section are alphabetized. Just the syntax of each item is given; refer to the item's reference template for a complete description.

Functions

Topic: Ignore

```
outcome api_accurate_bs3_approximation(FACE* face, double
   requested_tol, bs3_surface& fit_sur = *(bs3_surface*)NULL_REF,
   AcisOptions* ao = NULL)

outcome api_body(BODY*& bodyOut, AcisOptions* ao = NULL)

outcome api_body_mass_pr(BODY* body, SPAposition const&
   root_proj_pl, SPAunit_vector const& normal_proj_pl, int
   selector, double req_rel_accy, double& volume, SPAposition&
   cofg, tensor& inertia, double p_moments[], SPAunit_vector
   p_axes[], double& est_rel_accy_achieved, double
   sheet_thickness = 0.0, AcisOptions* ao = NULL)

outcome api_body_mass_pr(BODY* body, int selector, double
   req_rel_accy, double& volume, SPAposition& cofg, tensor&
   inertia, double p_moments[], SPAunit_vector p_axes[], double&
   est_rel_accy_achieved, double sheet_thickness = 0.0,
   AcisOptions* ao = NULL)

outcome api_body_to_2d(BODY* body, AcisOptions* ao = NULL)

outcome api_build_wire(BODY* given_body, logical closed, int
   length, SPAposition points[], curve* curves[], BODY*& body,
   AcisOptions* ao = NULL)

outcome api_closed_wire(BODY* body, AcisOptions* ao = NULL)
```

outcome **api_closed_wire**(WIRE* wire, AcisOptions* ao = NULL)

outcome **api_create_point**(const SPAposition& pos, APOINT*& pnt, AcisOptions* ao = NULL)

outcome **api_create_text**(const SPAposition& location, const char* string, const char* font, int size, TEXT_ENT*& text, AcisOptions* ao = NULL)

outcome **api_curve_arc**(const SPAposition& center, double radius, double start_angle, double end_angle, EDGE*& arc, AcisOptions* ao = NULL)

outcome **api_curve_arc_3curve**(const entity_with_ray& crv1, const entity_with_ray& crv2, const entity_with_ray& crv3, logical full, EDGE*& arc, AcisOptions* ao = NULL)

outcome **api_curve_arc_3pt**(const SPAposition& pt1, const SPAposition& pt2, const SPAposition& pt3, logical full, EDGE*& arc, AcisOptions* ao = NULL)

outcome **api_curve_arc_center_edge**(const SPAposition& center, const SPAposition& pt1, const SPAposition& pt2, const SPAunit_vector* norm, EDGE*& arc, AcisOptions* ao = NULL)

outcome **api_curve_arc_diagonal**(const SPAposition& pt1, const SPAposition& pt2, logical full, EDGE*& arc, AcisOptions* ao = NULL)

outcome **api_curve_bezier**(const SPAposition& pt1, const SPAposition& pt2, const SPAposition& pt3, const SPAposition& pt4, EDGE*& crv, AcisOptions* ao = NULL)

outcome **api_curve_ellipse**(const SPAposition& center, const SPAposition& major, double ratio, double start_angle, double end_angle, EDGE*& ell, AcisOptions* ao = NULL)

outcome **api_curve_fillet**(const entity_with_ray& crv1, const entity_with_ray& crv2, double radius, logical trim1, logical trim2, EDGE*& arc, AcisOptions* ao = NULL)

outcome **api_curve_law**(law* in_law, double start, double end, curve*& new_curve, int law_number = 0, law** other_laws = NULL, AcisOptions* ao = NULL)

outcome **api_curve_line**(const SPAposition& pt1, const SPAposition& pt2, EDGE*& line, AcisOptions* ao = NULL)

outcome **api_curve_line_tangent**(const SPAposition* pt1, const entity_with_ray* eray1, const SPAposition* pt2, const entity_with_ray* eray2, EDGE*& line, AcisOptions* ao = NULL)

```
outcome api_curve_spline(int numpts, const SPAposition* pts,
   const SPAunit_vector* start, const SPAunit_vector* end, EDGE*&
   crv, logical approx_ok = TRUE, logical periodic = FALSE,
   AcisOptions* ao = NULL)

outcome api_curve_spline2(int numpts, const SPAposition* pts,
   const double* params, const SPAvector* start, const SPAvector*
   end, EDGE*& crv, AcisOptions* ao = NULL)

outcome api_edge(EDGE* edgeIn, EDGE*& edgeOut, AcisOptions* ao =
   NULL)

outcome api_edge_arclength_metric(EDGE* edgeIn, double& metric,
   AcisOptions* ao = NULL)

outcome api_edge_arclength_param(EDGE* edgeIn, logical approx_ok,
   double tol, EDGE*& edgeOut, AcisOptions* ao = NULL)

outcome api_edge_law(law* inlaw, double start, double end, EDGE*&
   new_edge, int law_number = 0, law** other_laws = NULL,
   AcisOptions* ao = NULL)

outcome api_edge_plaw(FACE* in_face, law* inlaw, double start,
   double end, EDGE*& new_edge, int law_number = 0, law**
   other_laws = NULL, AcisOptions* ao = NULL)

outcome api_edge_spiral(SPAposition& center, SPAvector& normal,
   SPAposition& start_position, double width, double angle,
   EDGE*& spiral, logical handedness = TRUE, AcisOptions* ao =
   NULL)

outcome api_edge_spiral(SPAposition& center, SPAvector& normal,
   SPAvector& start_direction, double width, double angle, EDGE*&
   spiral, logical handedness = TRUE, double start_radius = -1.0,
   AcisOptions* ao = NULL)

outcome api_edge_spring(SPAposition& axis_point, SPAvector&
   axis_vector, SPAposition& start_position, logical handedness,
   int helix_count, double* thread_distance_array, double*
   rotation_angle_array, double* transition_height_array, double*
   transition_angle_array, EDGE*& crv, AcisOptions* ao = NULL)

outcome api_edge_spring_law(SPAposition& axis_point, SPAvector&
   axis_vector, SPAposition& start_position, law*
   variable_radius_law, logical handedness, int helix_count,
   double* thread_distance_array, double* rotation_angle_array,
   double* transition_height_array, double*
   transition_angle_array, EDGE*& crv, AcisOptions* ao = NULL)
```

```
outcome api_edge_spring_taper(SPAposition& axis_point, SPAvector&
   axis_vector, SPAposition& start_position, double taper_angle,
   logical handedness, int helix_count, double*
   thread_distance_array, double* rotation_angle_array, double*
   transition_height_array, double* transition_angle_array,
   EDGE*& crv, AcisOptions* ao = NULL)

outcome api_edge_to_spline(EDGE* e1, EDGE*& e2, AcisOptions* ao =
   NULL)

outcome api_ed_inters_to_ents(EDGE* e1, curve_curve_int* inters,
   ENTITY_LIST& ents, AcisOptions* ao = NULL)

outcome api_enclose_void(FACE* face, REVBIT sense, logical lumps,
   AcisOptions* ao = NULL)

outcome api_ent_area(ENTITY* ent, double req_rel_accy, double&
   area, double& est_rel_accy_achieved, AcisOptions* ao = NULL)

outcome api_face_conic(double radius, double conic_const, double
   extent, double length, FACE*& face, AcisOptions* ao = NULL)

outcome api_face_cylinder_cone(const SPAposition& center, const
   SPAvector& normal, double bottom, double top, double start,
   double end, double ratio, const SPAposition* pt, FACE*& face,
   AcisOptions* ao = NULL)

outcome api_face_law(law* in_law, double minu, double maxu,
   double minv, double maxv, FACE*& face, int in_law_number = 0,
   law** in_other_laws = NULL, AcisOptions* ao = NULL)

outcome api_face_plane(const SPAposition& p, double width, double
   height, const SPAvector* normal, FACE*& face, AcisOptions* ao
   = NULL)

outcome api_face_sphere(const SPAposition& center, double radius,
   double lo_start, double lo_end, double la_start, double
   la_end, const SPAvector* normal, FACE*& face, AcisOptions* ao
   = NULL)

outcome api_face_spl_apprx(const splgrid* grid, FACE*& face,
   AcisOptions* ao = NULL)

outcome api_face_spl_ctrlpts(const splsurf* surf, FACE*& face,
   AcisOptions* ao = NULL)

outcome api_face_spl_intrp(const splgrid* grid, FACE*& face,
   AcisOptions* ao = NULL)
```

```
outcome api_face_torus(const SPAposition& center, double major,
   double minor, double tu_start, double tu_end, double sv_start,
   double sv_end, const SPAvector* normal, FACE*& face,
   AcisOptions* ao = NULL)

outcome api_fillet_vertex(VERTEX* vert, double radius, EDGE*
   edge1 = NULL, EDGE* edge2 = NULL, AcisOptions* ao = NULL)

outcome api_find_face(BODY* body, SPAunit_vector const&
   direction, FACE*& face, AcisOptions* ao = NULL)

outcome api_find_vertex(BODY* body, SPAposition const& pos,
   VERTEX*& vertex, AcisOptions* ao = NULL)

outcome api_initialize_constructors()

outcome api_loop_external(LOOP* given_loop, logical* if_external,
   AcisOptions* ao = NULL)

outcome api_make_approx_curve(curve const* input_curve, double
   const tol, SPAinterval range, curve*& output_curve,
   AcisOptions* ao = NULL)

outcome api_make_approx_surface(surface const* input_surface,
   double const tol, SPAinterval const& u_range, SPAinterval
   const& v_range, surface*& output_surface, AcisOptions* ao =
   NULL)

outcome api_make_cnface(const SPAposition& center, const
   SPAunit_vector& normal_axis, const SPAvector& major_axis,
   double radius_ratio, double sint, double cost, double st_ang,
   double end_ang, double height, FACE*& face, AcisOptions* ao =
   NULL)

outcome api_make_cuboid(double width, double depth, double
   height, BODY*& body, AcisOptions* ao = NULL)

outcome api_make_edge_from_curve(curve const* cur, EDGE*& edge,
   AcisOptions* ao = NULL)

outcome api_make_ewire(int num_edges, EDGE* edges[], BODY*& body,
   AcisOptions* ao = NULL)

outcome api_make_ewires(int num_edges, EDGE* edges[], int&
   n_bodies, BODY**& bodies, AcisOptions* ao = NULL)

outcome api_make_frustum(double height, double rad1, double rad2,
   double top, BODY*& body, AcisOptions* ao = NULL)
```

```
outcome api_make_kwire(BODY* given_body, SPAunit_vector const&
   normal, int length_pts, SPAposition array_pts[], double
   array_bulges[], BODY*& body, AcisOptions* ao = NULL)

outcome api_make_planar_disk(const SPAposition& origin, const
   SPAunit_vector& normal, double radius, FACE*& face, logical
   half_space = FALSE, AcisOptions* ao = NULL)

outcome api_make_plface(const SPAposition& origin, const
   SPAposition& left, const SPAposition& right, FACE*& face,
   AcisOptions* ao = NULL)

outcome api_make_polygon(BODY*& polygon, SPAposition centre,
   SPAvector start, SPAvector& normal, double& side_length, int
   number_of_sides = 6, logical oncenter = FALSE, AcisOptions* ao
   = NULL)

outcome api_make_prism(double height, double rad1, double rad2,
   int nsides, BODY*& body, AcisOptions* ao = NULL)

outcome api_make_pyramid(double height, double rad1, double rad2,
   double top, int nsides, BODY*& body, AcisOptions* ao = NULL)

outcome api_make_spface(const SPAposition& center, double radius,
   const SPAunit_vector& lat, const SPAunit_vector& lon, double
   slat, double elat, double slon, double elon, FACE*& face,
   AcisOptions* ao = NULL)

outcome api_make_sphere(double radius, BODY*& body, AcisOptions*
   ao = NULL)

outcome api_make_spline(spline const& this_spline, BODY*& body,
   AcisOptions* ao = NULL)

outcome api_make_torus(double major_radius, double minor_radius,
   BODY*& body, AcisOptions* ao = NULL)

outcome api_make_trface(const SPAposition& center, const
   SPAunit_vector& normal, double major, double minor, const
   SPAposition& pnt, double uf, double ut, double vf, double vt,
   FACE*& face, AcisOptions* ao = NULL)

outcome api_make_wire(BODY* given_body, int length_pts,
   SPAposition array_pts[], BODY*& body, AcisOptions* ao = NULL)

outcome api_manifold_class(BODY* given_body, ENTITY_LIST*&
   mani_shells, ENTITY_LIST*& mani_sheets, ENTITY_LIST&
   wire_edges, ENTITY_LIST& lamina_faces, ENTITY_LIST*&
   edge_arcs, ENTITY_LIST*& vertex_arcs, AcisOptions* ao = NULL)
```

outcome **api_mk_by_faces**(BODY* given_body, int num_faces, FACE* faces[], BODY*& body, AcisOptions* ao = NULL)

outcome **api_mk_ed_bs3_curve**(bs3_curve bs, EDGE*& edge, AcisOptions* ao = NULL)

outcome **api_mk_ed_conic**(const SPAposition& start, const SPAunit_vector& start_tan, const SPAposition& end, const SPAunit_vector& end_tan, double rho, EDGE*& edge, AcisOptions* ao = NULL)

outcome **api_mk_ed_cubic**(int npts, const SPAposition pts[], SPAunit_vector const& start_dir, SPAunit_vector const& end_dir, double fitol, EDGE*& edge, AcisOptions* ao = NULL)

outcome **api_mk_ed_ellipse**(SPAposition const& center, SPAunit_vector const& normal, SPAvector const& major_axis, double radius_ratio, double start_angle, double end_angle, EDGE*& edge, AcisOptions* ao = NULL)

outcome **api_mk_ed_int_ctrlpts**(int degree, logical rational, logical closed, logical periodic, int num_ctrlpts, const SPAposition ctrlpts[], const double weights[], double point_tol, int num_knots, const double knots[], double knot_tol, EDGE*& edge, AcisOptions* ao = NULL)

outcome **api_mk_ed_line**(SPAposition const& start_point, SPAposition const& end_point, EDGE*& edge, AcisOptions* ao = NULL)

outcome **api_mk_fa_spl_ctrlpts**(int degree_u, logical rational_u, int form_u, int pole_u, int num_ctrlpts_u, int degree_v, logical rational_v, int form_v, int pole_v, int num_ctrlpts_v, const SPAposition ctrlpts[], const double weights[], double point_tol, int num_knots_u, const double knots_u[], int num_knots_v, const double knots_v[], double knot_tol, FACE*& face, AcisOptions* ao = NULL)

outcome **api_mk_fa_spl_fit**(double fittol, int num_pts_u, int num_pts_v, const SPAposition pts[], const SPAunit_vector du_s[], const SPAunit_vector du_e[], FACE*& face, AcisOptions* ao = NULL)

outcome **api_mk_fa_spl_intp**(int num_pts_u, int num_pts_v, const SPAposition pts[], const SPAunit_vector du_s[], const SPAunit_vector du_e[], const SPAunit_vector dv_s[], const SPAunit_vector dv_e[], FACE*& face, AcisOptions* ao = NULL)

```
outcome api_modify_ellipse(EDGE* ell, const SPAposition& center,
   const SPAunit_vector& normal, const SPAvector& major_axis,
   double radius_ratio, double start_angle, double end_angle,
   AcisOptions* ao = NULL)

outcome api_modify_line(EDGE* line, const SPAposition& st_pt,
   const SPAposition& end_pt, AcisOptions* ao = NULL)

outcome api_planar_face_pr(FACE* face, double req_rel_accy,
   double& area, SPAposition& centre, double& moment_a, double&
   moment_b, SPAunit_vector& axis_a, SPAunit_vector& axis_b,
   double& est_rel_accy_achieved, AcisOptions* ao = NULL)

outcome api_q_edges_around_vertex(VERTEX* vertex, ENTITY_LIST*
   edge_list, AcisOptions* ao = NULL)

outcome api_reverse_body(BODY* body, AcisOptions* ao = NULL)

outcome api_reverse_face(FACE*& face, AcisOptions* ao = NULL)

outcome api_reverse_wire(ENTITY* in_body, AcisOptions* ao = NULL)

outcome api_reverse_wire(WIRE* wire, AcisOptions* ao = NULL)

outcome api_sheet_from_ff(int num_faces, FACE* faces[], BODY*&
   new_body, AcisOptions* ao = NULL)

outcome api_shell_external(SHELL* given_shell, int& external,
   AcisOptions* ao = NULL)

outcome api_solid_block(const SPAposition& pt1, const
   SPAposition& pt2, BODY*& block, AcisOptions* ao = NULL)

outcome api_solid_cylinder_cone(const SPAposition& pt1, const
   SPAposition& pt2, double major_radius, double minor_radius,
   double top_radius, const SPAposition* xpt, BODY*& cyl_or_cone,
   AcisOptions* ao = NULL)

outcome api_solid_sphere(const SPAposition& center, double
   radius, BODY*& sph, AcisOptions* ao = NULL)

outcome api_solid_torus(const SPAposition& center, double
   major_radius, double minor_radius, BODY*& tor, AcisOptions* ao
   = NULL)

outcome api_split_curve(EDGE* crv, const SPAposition* split_pt,
   EDGE* split_crv, ENTITY_LIST& new_crvs, AcisOptions* ao =
   NULL)
```

```
outcome api_split_edge_at_disc(EDGE* theEdge, ENTITY_LIST&
   new_edges, int cont_order = 1, AcisOptions* ao = NULL)

outcome api_terminate_constructors()

outcome api_trans_edge(EDGE* edgeIn, SPAtransf const& t, EDGE*&
   edgeOut, AcisOptions* ao = NULL)

outcome api_trim_2curves(const entity_with_ray& crv1, const
   entity_with_ray& crv2, AcisOptions* ao = NULL)

outcome api_trim_chain(int num_crv, const entity_with_ray* crvs,
   logical close, ENTITY_LIST& trimmed_crvs, AcisOptions* ao =
   NULL)

outcome api_trim_curve(const entity_with_ray& eray1, const
   SPAposition* trim_pt, const entity_with_ray* eray2,
   AcisOptions* ao = NULL)

outcome api_trim_middle(const entity_with_ray& crv, const
   SPAposition* trim_pt1, const entity_with_ray* trimr1, const
   SPAposition* trim_pt2, const entity_with_ray* trimr2, EDGE*&
   new_edge, AcisOptions* ao = NULL)

outcome api_wiggle(double width, double depth, double height, int
   low_v_type, int high_v_type, int low_u_type, int high_u_type,
   logical height_given, BODY*& bodyOut, AcisOptions* ao = NULL)

outcome api_wire_len(BODY* body, double& length, AcisOptions* ao
   = NULL)

outcome api_wire_len(WIRE* wire, double& length, AcisOptions* ao
   = NULL)

outcome api_wire_to_chain(BODY* body, ENTITY_LIST& list,
   AcisOptions* ao = NULL)

logical is_PRIMITIVE_ANNOTATION(const ENTITY* e)

FACE* make_face_spline(surface const& this_surface, curve const&
   bottom_cur = *(curve const*)NULL_REF, curve const& top_cur =
   *(curve const*)NULL_REF, curve const& left_cur = *(curve
   const*)NULL_REF, curve const& right_cur = *(curve
   const*)NULL_REF, SPApar_box const& pb = *(SPApar_box
   const*)NULL_REF)
```

Scheme Extensions

Topic: Ignore

```
(arc:set-center curve center)
```

```
(arc:set-direction curve vector)

(arc:set-end curve angle)

(arc:set-normal curve vector)

(arc:set-radius curve radius)

(arc:set-radius-ratio curve ratio)

(arc:set-start curve angle)

(body:find-face body direction)

(body:reverse body)

(coedge:add-pcurve coedge)

(curve:circular center radius [normal=z-axis])

(curve:elliptical center major-axis [normal=z-axis]
    [ratio])

(curve:linear pos-start pos-gvec)

(edge:arclength-metric edge)

(edge:arclength-param edge [approx-ok])

(edge:bezier point1 point2 point3 point4)

(edge:bezier-ndeg (list-points))

(edge:circular center-position radius
    [start-angle=0 [end-angle=360]])

(edge:circular-3curve entray1 entray2 entray3
    [full=#t])

(edge:circular-3pt edge-pos1 edge-pos2
    edge-pos3 [full=#f])

(edge:circular-center-rim center-pos
    edge-pos1 [edge-pos2=360])

(edge:circular-diameter edge-pos1
    edge-pos2 [full=#f])

(edge:conic point1 point2 point3 rho)

(edge:ellipse center normal major-axis
    [radius-ratio=1 [start-angle=0 end-angle=360]])
```

```
(edge:elliptical center–pos start–pos ratio
    [start–angle=0 [end–angle=360]])

(edge:end edge)

(edge:end–dir edge)

(edge:extend in–entity in–start in–end)

(edge:fillet entray1 entray2 radius
    [trim1=#t [trim2=#t]])

(edge:from–curve curve [start end])

(edge:get–tolerance edge [force–update])

(edge:law law start end [perp–law])

(edge:length edge)

(edge:linear {position | point | entray}
    {position | point | entray})

(edge:mid–point edge [approximation=#t])

(edge:mid–point–dir edge [approximation=#t])

(edge:plaw surface law start end)

(edge:reverse edge)

(edge:spiral center normal [start–position |
    start–direction] width angle [handedness
    start–radius])

(edge:spline position–list
    [[start–angle | start–direction | spline–condition]
    [end–angle | end–direction]] ["periodic"]
    ["exact"])

(edge:spline position–list param–list
[start–direction | spline–condition]
[end–direction])

(edge:split edge1 {position | edge2})

(edge:split–at–disc edge [disc–order | acis–opts])

(edge:spring axis–point axis–vector
    start–position handedness thread–distance
    rotation–degrees {[trans–height trans–degrees
    thread–distance rotation–degrees]...})
```

```
(edge:spring–law axis–point axis–vector
    start–position radius–law handedness
    thread–distance rotation–degrees
    {[trans–height trans–degrees
    thread–distance rotation–degrees]...})

(edge:spring–tapte axis–point axis–vector start–position
    taper–angle handedness thread–distance
    rotation–degrees {[trans–height trans–degrees
    thread–distance rotation–degrees]...})

(edge:start edge)

(edge:start–dir edge)

(edge:tolerant edge)

(edge:trim entray {position | entray})

(edge:trim–chain entray–list close)

(edge:trim–intersect entray entray)

(edge:trim–middle entray {position | entray}
    {position | entray})

(entity:accurate–approx entity [fit=resfit])

(entity:box entity–list [create–box])

(entity:tolerize entity)

(face:area face)

(face:cone position1 position2 radius1 radius2
    [start–angle=0 end–angle=360 ratio position3])

(face:conic radius conic–constant extent [length])

(face:cylinder position1 position2 radius
    [start–angle=0 end–angle=360 ratio position3])

(face:extend entity start–u end–u start–v end–v)

(face:law law start–u end–u start–v end–v)

(face:par–box face)

(face:planar–disk center normal radius)
```

```
(face:plane position width height [normal])

(face:prop face [tolerance=0.01])

(face:reverse face)

(face:sphere position radius
    [latitude-start=-90 [latitude-end=90
    [longitude-start=0 [longitude-end=360 [poledir]]]]])

(face:spline-ctrlpts surface-data)

(face:spline-grid grid-data interpolate)

(face:torus position spine-radius tube-radius
    [tube-start=0 tube-end=360 spine-start=0
    spine-end=360 vector])

(line:set-direction curve direction)

(line:set-end curve position)

(line:set-start curve position)

(loop:external? loop)

(point position)

(sheet:1d body)

(sheet:2d body)

(sheet:enclose face side [lumps=#f])

(sheet:face face)

(solid:area solid-body [tolerance=0.01])

(solid:block {{diagonal1 diagonal2 | (diagonal-x1
    diagonal-y1 diagonal-z1 diagonal-x2 diagonal-y2
    diagonal-z2)}})

(solid:closed? object)

(solid:cone {{start-axis end-axis} | {start-axis-x
    start-axis-y start-axis-z end-axis-x end-axis-y
    end-axis-z}} base-radius top-radius
    [ratio=1 [position3 | {x3 y3 z3}]])

(solid:cylinder {{start-pos end-pos} | {x-start
    y-start z-start x-end y-end z-end}} radius
    [ratio=1 {position3 | {x3 y3 z3}}])
```

(**solid:manifold?** object)

(**solid:massprop** entity [integer–type=0 [thickness] tolerance=0.01
{{position=center} | {x=center–x y=center–y
z=center–z}} direction=z–axis])

(**solid:prism** height major–radius minor–radius nsides)

(**solid:pyramid** height major–radius minor–radius nsides
[top])

(**solid:sphere** {center–position | {center–x center–y
center–z}} radius)

(**solid:torus** {center–position | {center–x center–y
center–z}} major–radius minor–radius)

(**solid:wiggle** width depth height
{wiggle–type | shape1 shape2 shape3 shape4})

(**splgrid**)

(**splgrid:copy** grid)

(**splgrid:point–item** grid row column)

(**splgrid:print** grid)

(**splgrid:set–point–item** grid row column value)

(**splgrid:set–point–list** grid list rows columns)

(**splgrid:set–tolerance** grid value)

(**splgrid:set–u–tanvec–item** grid index value
start–or–end)

(**splgrid:set–u–tanvec–list** grid list start–or–end)

(**splgrid:set–v–tanvec–item** grid index value
start–or–end)

(**splgrid:set–v–tanvec–list** grid list start–or–end)

(**splgrid:tolerance** grid)

(**splgrid:u–points** grid)

(**splgrid:u–tanvec–item** grid index start–or–end)

(**splgrid:v–points** grid)

(**splgrid:v-tanvec-item** grid index start-or-end)

(**splgrid?** object)

(**splsurf**)

(**splsurf:copy** surface)

(**splsurf:ctrlpt-count** surface)

(**splsurf:ctrlpt-item** surface row column)

(**splsurf:param** surface param-str)

(**splsurf:print** surface)

(**splsurf:set-ctrlpt-item** surface row column value)

(**splsurf:set-ctrlpt-list** surface list row column)

(**splsurf:set-u-knot-item** surface index value)

(**splsurf:set-u-knot-list** surface list count)

(**splsurf:set-u-param** surface degree rational form pole)

(**splsurf:set-v-knot-item** surface index value)

(**splsurf:set-v-knot-list** surface list count)

(**splsurf:set-v-param** surface degree rational form pole)

(**splsurf:set-weight-item** surface row column value)

(**splsurf:set-weight-list** surface list)

(**splsurf:u-knot-count** surface)

(**splsurf:u-knot-item** surface index)

(**splsurf:v-knot-count** surface)

(**splsurf:v-knot-item** surface index)

(**splsurf:weight-item** surface row column)

(**splsurf?** object)

(**text** position text-string [font] [size])

(**tolerant:fix** entity)

(**tolerant:move** edge vector)

(**tolerant:none** entity)

(**tolerant:optimize** entity)

(**tolerant:optimized?** check–entity)

(**tolerant:report** entity)

(**tolerant:update** entity)

(**tolerant?** check–entity)

(**vertex:fillet** vertex radius [edge1 edge2])

(**vertex:get–tolerance** vertex [force–update])

(**vertex:tolerant** vertex)

(**wire–body** entity–list)

(**wire–body:kwire** [vector] position {bulge position}*)

(**wire–body:length** wire)

(**wire–body:points** plist)

(**wire–body:polygon** center start normal
    side–length sides on–center output–string)

(**wire–body:unbranch** entity–list)

(**wire:end** wire)

(**wire:reverse** wire)

(**wire:start** wire)