Skip to content

Curves Operators

bpy.ops.curves.**add_bezier(*, radius=1.0, enter_editmode=False, align='WORLD', location=(0.0, 0.0, 0.0), rotation=(0.0, 0.0, 0.0), scale=(0.0, 0.0, 0.0))**

Add new bezier curve

PARAMETERS:

- **radius** (*float in [0, inf], (optional)*) – Radius
- **enter_editmode** (*boolean, (optional)*) – Enter Edit Mode, Enter edit mode when adding this object
- **align** (*enum in ['WORLD', 'VIEW', 'CURSOR'], (optional)*) –
  Align, The alignment of the new object
  - `WORLD` World – Align the new object to the world.
  - `VIEW` View – Align the new object to the view.
  - `CURSOR` 3D Cursor – Use the 3D cursor orientation for the new object.
- **location** (`mathutils.Vector` of 3 items in [-inf, inf], (optional)) – Location, Location for the newly added object
- **rotation** (`mathutils.Euler` rotation of 3 items in [-inf, inf], (optional)) – Rotation, Rotation for the newly added object
- **scale** (`mathutils.Vector` of 3 items in [-inf, inf], (optional)) – Scale, Scale for the newly added object

bpy.ops.curves.**add_circle(*, radius=1.0, enter_editmode=False, align='WORLD', location=(0.0, 0.0, 0.0), rotation=(0.0, 0.0, 0.0), scale=(0.0, 0.0, 0.0))**

Add new circle curve

PARAMETERS:

- **radius** (*float in [0, inf], (optional)*) – Radius
- **enter_editmode** (*boolean, (optional)*) – Enter Edit Mode, Enter edit mode when adding this object
- **align** (*enum in ['WORLD', 'VIEW', 'CURSOR'], (optional)*) –
  Align, The alignment of the new object
  - `WORLD` World – Align the new object to the world.
  - `VIEW` View – Align the new object to the view.
  - `CURSOR` 3D Cursor – Use the 3D cursor orientation for the new object.
- **location** (`mathutils.Vector` of 3 items in [-inf, inf], (optional)) – Location, Location for the newly added object
- **rotation** (`mathutils.Euler` rotation of 3 items in [-inf, inf], (optional)) – Rotation, Rotation for the newly added object
- **scale** (`mathutils.Vector` of 3 items in [-inf, inf], (optional)) – Scale, Scale for the newly added object

bpy.ops.curves.**attribute_set(*, value_float=0.0, value_float_vector_2d=(0.0, 0.0), value_float_vector_3d=(0.0, 0.0, 0.0), value_int=0, value_int_vector_2d=(0, 0), value_color=(1.0, 1.0, 1.0, 1.0), value_bool=False)**

Set values of the active attribute for selected elements

PARAMETERS:

- **value_float** (*float in [-inf, inf], (optional)*) – Value
- **value_float_vector_2d** (*float array of 2 items in [-inf, inf], (optional)*) – Value
- **value_float_vector_3d** (*float array of 3 items in [-inf, inf], (optional)*) – Value
- **value_int** (*int in [-inf, inf], (optional)*) – Value
- **value_int_vector_2d** (*int array of 2 items in [-inf, inf], (optional)*) – Value
- **value_color** (*float array of 4 items in [-inf, inf], (optional)*) – Value
- **value_bool** (*boolean, (optional)*) – Value

bpy.ops.curves.**convert_from_particle_system()**

Add a new curves object based on the current state of the particle system

bpy.ops.curves.**convert_to_particle_system()**

Add a new or update an existing hair particle system on the surface object

bpy.ops.curves.**curve_type_set(*, type='POLY', use_handles=False)**

Set type of selected curves

PARAMETERS:

- **type** (enum in Curves Type Items, (optional)) – Type, Curve type
- **use_handles** (*boolean, (optional)*) – Handles, Take handle information into account in the conversion

bpy.ops.curves.**cyclic_toggle()**

Make active curve closed/opened loop

bpy.ops.curves.**delete()**

Remove selected control points or curves

bpy.ops.curves.**draw(*, error_threshold=0.0, fit_method='REFIT', corner_angle=1.22173, use_cyclic=True, stroke=None, wait_for_input=True, is_curve_2d=False, bezier_as_nurbs=False)**

Draw a freehand curve

PARAMETERS:

- **error_threshold** (*float in [0, 10], (optional)*) – Error, Error distance threshold (in object units)
- **fit_method** (enum in Curve Fit Method Items, (optional)) – Fit Method
- **corner_angle** (*float in [0, 3.14159], (optional)*) – Corner Angle
- **use_cyclic** (*boolean, (optional)*) – Cyclic
- **stroke** (`bpy_prop_collection` of `OperatorStrokeElement`, (optional)) – Stroke
- **wait_for_input** (*boolean, (optional)*) – Wait for Input
- **is_curve_2d** (*boolean, (optional)*) – Curve 2D
- **bezier_as_nurbs** (*boolean, (optional)*) – As NURBS

bpy.ops.curves.**duplicate()**

Copy selected points or curves

bpy.ops.curves.**duplicate_move(*, CURVES_OT_duplicate=None, TRANSFORM_OT_translate=None)**

Make copies of selected elements and move them

PARAMETERS:

- **CURVES_OT_duplicate** (`CURVES_OT_duplicate`, (optional)) – Duplicate, Copy selected points or curves
- **TRANSFORM_OT_translate** (`TRANSFORM_OT_translate`, (optional)) – Move, Move selected items

bpy.ops.curves.**extrude()**

Extrude selected control point(s)

bpy.ops.curves.**extrude_move(*, CURVES_OT_extrude=None, TRANSFORM_OT_translate=None)**

Extrude curve and move result

PARAMETERS:

- **CURVES_OT_extrude** (`CURVES_OT_extrude`, (optional)) – Extrude, Extrude selected control point(s)
- **TRANSFORM_OT_translate** (`TRANSFORM_OT_translate`, (optional)) – Move, Move selected items

bpy.ops.curves.**handle_type_set(*, type='AUTO')**

Set the handle type for bezier curves

PARAMETERS:

type (enum in Curves Handle Type Items, (optional)) – Type

bpy.ops.curves.**sculptmode_toggle()**

Enter/Exit sculpt mode for curves

bpy.ops.curves.**select_all(*, action='TOGGLE')**

(De)select all control points

PARAMETERS:

action (*enum in ['TOGGLE', 'SELECT', 'DESELECT', 'INVERT'], (optional)*) –

Action, Selection action to execute

- `TOGGLE` Toggle – Toggle selection for all elements.
- `SELECT` Select – Select all elements.
- `DESELECT` Deselect – Deselect all elements.
- `INVERT` Invert – Invert selection of all elements.

bpy.ops.curves.**select_ends(*, amount_start=0, amount_end=1)**

Select end points of curves

PARAMETERS:

- **amount_start** (*int in [0, inf], (optional)*) – Amount Front, Number of points to select from the front
- **amount_end** (*int in [0, inf], (optional)*) – Amount Back, Number of points to select from the back

bpy.ops.curves.**select_less()**

Shrink the selection by one point

bpy.ops.curves.**select_linked()**

Select all points in curves with any point selection

bpy.ops.curves.**select_linked_pick(*, deselect=False)**

Select all points in the curve under the cursor

PARAMETERS:

deselect (*boolean, (optional)*) – Deselect, Deselect linked control points rather than selecting them

bpy.ops.curves.**select_more()**

Grow the selection by one point

bpy.ops.curves.**select_random(*, seed=0, probability=0.5)**

Randomizes existing selection or create new random selection

PARAMETERS:

- **seed** (*int in [-inf, inf], (optional)*) – Seed, Source of randomness
- **probability** (*float in [0, 1], (optional)*) – Probability, Chance of every point or curve being included in the selection

bpy.ops.curves.**set_selection_domain(*, domain='POINT')**

Change the mode used for selection masking in curves sculpt mode

PARAMETERS:

domain (enum in Attribute Curves Domain Items, (optional)) – Domain

bpy.ops.curves.**snap_curves_to_surface(*, attach_mode='NEAREST')**

Move curves so that the first point is exactly on the surface mesh

PARAMETERS:

attach_mode (*enum in ['NEAREST', 'DEFORM'], (optional)*)

attach_mode (*enum in ['NEAREST', 'DEFORM'], (optional)*) –

Attach Mode, How to find the point on the surface to attach to

- `NEAREST` Nearest – Find the closest point on the surface for the root point of every curve and move the root there.
- `DEFORM` Deform – Re-attach curves to a deformed surface using the existing attachment information. This only works when the topology of the surface mesh has not changed.

bpy.ops.curves.**subdivide(*, number_cuts=1)**

Subdivide selected curve segments

PARAMETERS:

number_cuts (*int in [1, 1000], (optional)*) – Number of Cuts

bpy.ops.curves.**surface_set()**

Use the active object as surface for selected curves objects and set it as the parent

bpy.ops.curves.**switch_direction()**

Reverse the direction of the selected curves

bpy.ops.curves.**tilt_clear()**

Clear the tilt of selected control points