Skip to content

SpaceNodeEditor(Space)

base classes — bpy_struct, Space

class bpy.types.**SpaceNodeEditor(Space)**

Node editor space data

backdrop_channels

Channels of the image to draw

- COLOR_ALPHA Color & Alpha – Display image with RGB colors and alpha transparency.
- COLOR Color – Display image with RGB colors.
- ALPHA Alpha – Display alpha transparency channel.
- RED Red.
- GREEN Green.
- BLUE Blue.

TYPE:

enum in ['COLOR_ALPHA', 'COLOR', 'ALPHA', 'RED', 'GREEN', 'BLUE'], default 'COLOR'

backdrop_offset

Backdrop offset

TYPE:

float array of 2 items in [-inf, inf], default (0.0, 0.0)

backdrop_zoom

Backdrop zoom factor

TYPE:

float in [0.01, inf], default 1.0

cursor_location

Location for adding new nodes

TYPE:

mathutils.Vector of 2 items in [-inf, inf], default (0.0, 0.0)

edit_tree

Node tree being displayed and edited

TYPE:

NodeTree, (readonly)

geometry_nodes_tool_tree

Node group to edit as node tool

TYPE:

NodeTree

geometry_nodes_type

- MODIFIER Modifier – Edit node group from active object's active modifier.
- TOOL Tool – Edit any geometry node group for use as an operator.

TYPE:

enum in [‘MODIFIER’, ‘TOOL’], default ‘MODIFIER’

id

Data-block whose nodes are being edited

TYPE:

ID, (readonly)

id_from

Data-block from which the edited data-block is linked

TYPE:

ID, (readonly)

insert_offset_direction

Direction to offset nodes on insertion

TYPE:

enum in [‘RIGHT’, ‘LEFT’], default ‘RIGHT’

node_tree

Base node tree from context

TYPE:

NodeTree

overlay

Settings for display of overlays in the Node Editor

TYPE:

SpaceNodeOverlay, (readonly, never None)

path

Path from the data-block to the currently edited node tree

TYPE:

SpaceNodeEditorPath bpy_prop_collection of NodeTreePath, (readonly)

pin

Use the pinned node tree

TYPE:

boolean, default False

shader_type

Type of data to take shader from

- OBJECT Object – Edit shader nodes from Object.
- WORLD World – Edit shader nodes from World.
- LINESTYLE Line Style – Edit shader nodes from Line Style.

TYPE:

enum in [‘OBJECT’, ‘WORLD’, ‘LINESTYLE’], default ‘OBJECT’

show_annotation

Show annotations for this view

TYPE:

boolean, default False

show_backdrop

Use active Viewer Node output as backdrop for compositing nodes

TYPE:

boolean, default False

show_region_toolbar

TYPE:

boolean, default False

show_region_ui

TYPE:

boolean, default False

supports_previews

Whether the node editor's type supports displaying node previews

TYPE:

boolean, default False, (readonly)

texture_type

Type of data to take texture from

- `WORLD` World – Edit texture nodes from World.
- `BRUSH` Brush – Edit texture nodes from Brush.
- `LINESTYLE` Line Style – Edit texture nodes from Line Style.

TYPE:

enum in ['WORLD', 'BRUSH', 'LINESTYLE'], default 'WORLD'

tree_type

Node tree type to display and edit

TYPE:

enum in ['DUMMY'], default 'DUMMY'

cursor_location_from_region(x, y)

Set the cursor location using region coordinates

PARAMETERS:

- **x** (*int in [-inf, inf]*) – x, Region x coordinate
- **y** (*int in [-inf, inf]*) – y, Region y coordinate

classmethod bl_rna_get_subclass(id, default=None)

PARAMETERS:

id (*str*) – The RNA type identifier.

RETURNS:

The RNA type or default when not found.

RETURN TYPE:

`bpy.types.Struct` subclass

classmethod bl_rna_get_subclass_py(id, default=None)

PARAMETERS:

id (*str*) – The RNA type identifier.

RETURNS:

The class or default when not found.

RETURN TYPE:

type

classmethod draw_handler_add(callback, args, region_type, draw_type)

Add a new draw handler to this space type. It will be called every time the specified region in the space type will be drawn. Note: All argumer are positional only for now.

PARAMETERS:

- **callback** (*Callable[[Any, ...], Any]*) – A function that will be called when the region is drawn. It gets the specified arguments as input, it's return value is ignored.
- **args** (*tuple[Any, ...]*) – Arguments that will be passed to the callback.
- **region_type** (*str*) – The region type the callback draws in; usually `WINDOW`. (`bpy.types.Region.type`)
- **draw_type** (*str*) – Usually `POST_PIXEL` for 2D drawing and `POST_VIEW` for 3D drawing. In some cases `PRE_VIEW` can be used. `BACKDROP` can be used for backdrops in the node editor.

RETURNS:

Handler that can be removed later on.

RETURN TYPE:

object

classmethod draw_handler_remove(handler, region_type)

Remove a draw handler that was added previously.

PARAMETERS:

- **handler** (*object*) – The draw handler that should be removed.
- **region_type** (*str*) – Region type the callback was added to.

Inherited Properties

- `bpy_struct.id_data`
- `Space.show_locked_time`
- `Space.type`
- `Space.show_region_header`

Inherited Functions

- `bpy_struct.as_pointer`
- `bpy_struct.driver_add`
- `bpy_struct.driver_remove`
- `bpy_struct.get`
- `bpy_struct.id_properties_clear`
- `bpy_struct.id_properties_ensure`
- `bpy_struct.id_properties_ui`
- `bpy_struct.is_property_hidden`
- `bpy_struct.is_property_overridable_library`
- `bpy_struct.is_property_readonly`
- `bpy_struct.is_property_set`
- `bpy_struct.items`
- `bpy_struct.keyframe_delete`
- `bpy_struct.keyframe_insert`
- `bpy_struct.keys`
- `bpy_struct.path_from_id`
- `bpy_struct.path_resolve`
- `bpy_struct.pop`
- `bpy_struct.property_overridable_library_set`
- `bpy_struct.property_unset`
- `bpy_struct.type_recast`
- `bpy_struct.values`
- `Space.bl_rna_get_subclass`
- `Space.bl_rna_get_subclass_py`
- `Space.draw_handler_add`
- `Space.draw_handler_remove`