Skip to content

Context(bpy_struct)

base class — bpy_struct

class bpy.types.**Context(bpy_struct)**

Current windowmanager and data context

area

TYPE:

Area, (readonly)

asset

TYPE:

AssetRepresentation, (readonly)

blend_data

TYPE:

BlendData, (readonly)

collection

TYPE:

Collection, (readonly)

engine

TYPE:

string, default “”, (readonly, never None)

gizmo_group

TYPE:

GizmoGroup, (readonly)

layer_collection

TYPE:

LayerCollection, (readonly)

mode

TYPE:

enum in Context Mode Items, default ‘EDIT_MESH’, (readonly)

preferences

TYPE:

Preferences, (readonly)

region

TYPE:

Region, (readonly)

region_data

TYPE:

RegionView3D, (readonly)

region_popup

The temporary region for pop-ups (including menus and pop-overs)

TYPE:

Region, (readonly)

scene

TYPE:

Scene, (readonly)

screen

TYPE:

Screen, (readonly)

space_data

The current space, may be None in background-mode, when the cursor is outside the window or when using menu-search

TYPE:

Space, (readonly)

tool_settings

TYPE:

ToolSettings, (readonly)

view_layer

TYPE:

ViewLayer, (readonly)

window

TYPE:

Window, (readonly)

window_manager

TYPE:

WindowManager, (readonly)

workspace

TYPE:

WorkSpace, (readonly)

evaluated_depsgraph_get()

Get the dependency graph for the current scene and view layer, to access to data-blocks with animation and modifiers applied. If any data-blocks have been edited, the dependency graph will be updated. This invalidates all references to evaluated data-blocks from the dependency graph.

RETURNS:

Evaluated dependency graph

RETURN TYPE:

Depsgraph

copy()

path_resolve(path, coerce=True)

Returns the property from the path, raise an exception when not found.

PARAMETERS:

- **path** (*str*) – patch which this property resolves.
- **coerce** (*bool*) – optional argument, when True, the property will be converted into its Python representation.

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

temp_override(*, window=None, area=None, region=None, **keywords)

Context manager to temporarily override members in the context.

PARAMETERS:

- **window** (`bpy.types.Window`) – Window override or None.
- **screen** (`bpy.types.Screen`) –

  Screen override or None.

  Note

  Switching to or away from full-screen areas & temporary screens isn't supported. Passing in these screens will raise an exception, actions that leave the context such screens won't restore the prior screen.

  Note

  Changing the screen has wider implications than other arguments as it will also change the works-space and potentially the scene (when pinned).

- **area** (`bpy.types.Area`) – Area override or None.
- **region** (`bpy.types.Region`) – Region override or None.
- **keywords** – Additional keywords override context members.

RETURNS:

The context manager .

RETURN TYPE:

ContextTempOverride

Overriding the context can be used to temporarily activate another `window` / `area` & `region`, as well as other members such as the `active_object` or `bone`.

Notes:

- When overriding window, area and regions: the arguments must be consistent, so any region argument that's passed in must be contained b the current area or the area passed in. The same goes for the area needing to be contained in the current window.
- Temporary context overrides may be nested, when this is done, members will be added to the existing overrides.
- Context members are restored outside the scope of the context manager. The only exception to this is when the data is no longer available

- Context members are restored outside the scope of the context-manager. The only exception to this is when the data is no longer available In the event windowing data was removed (for example), the state of the context is left as-is. While this isn't likely to happen, explicit window operation such as closing windows or loading a new file remove the windowing data that was set before the temporary context wa created.

Overriding the context can be useful to set the context after loading files (which would otherwise by None). For example:

```
import bpy
from bpy import context

# Reload the current file and select all.
bpy.ops.wm.open_mainfile(filepath=bpy.data.filepath)
window = context.window_manager.windows[0]
with context.temp_override(window=window):
    bpy.ops.mesh.primitive_uv_sphere_add()
    # The context override is needed so it's possible to set edit-mode.
    bpy.ops.object.mode_set(mode='EDIT')
```

This example shows how it's possible to add an object to the scene in another window.

```
import bpy
from bpy import context

win_active = context.window
win_other = None
for win_iter in context.window_manager.windows:
    if win_iter != win_active:
        win_other = win_iter
        break

# Add cube in the other window.
with context.temp_override(window=win_other):
    bpy.ops.mesh.primitive_cube_add()
```

Inherited Properties

- bpy_struct.id_data

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

References

- AssetShelf.draw_context_menu
- AssetShelf.poll
- FileHandler.poll_drop
- Gizmo.draw
- Gizmo.draw_select
- Gizmo.exit
- Gizmo.invoke
- Gizmo.modal
- Gizmo.test_select
- GizmoGroup.draw_prepare
- GizmoGroup.invoke_prepare
- GizmoGroup.poll
- GizmoGroup.refresh
- GizmoGroup.setup
- Header.draw
- KeyingSetInfo.generate
- KeyingSetInfo.iterator
- KeyingSetInfo.poll
- Macro.draw
- Macro.poll
- Menu.draw
- Menu.poll
- Node.draw_buttons
- Node.draw_buttons_ext
- Node.init
- Node.socket_value_update
- NodeInternal.draw_buttons
- NodeInternal.draw_buttons_ext
- NodeSocket.draw
- NodeSocket.draw_color
- NodeSocketStandard.draw
- NodeSocketStandard.draw_color
- NodeTree.get_from_context
- NodeTree.interface_update
- NodeTree.poll
- NodeTreeInterfaceSocket.draw
- NodeTreeInterfaceSocketBool.draw
- NodeTreeInterfaceSocketCollection.draw
- NodeTreeInterfaceSocketColor.draw
- NodeTreeInterfaceSocketFloat.draw
- NodeTreeInterfaceSocketFloatAngle.draw
- NodeTreeInterfaceSocketFloatColorTemperature.draw
- NodeTreeInterfaceSocketFloatDistance.draw
- NodeTreeInterfaceSocketFloatFactor.draw
- NodeTreeInterfaceSocketFloatFrequency.draw
- NodeTreeInterfaceSocketInt.draw
- NodeTreeInterfaceSocketIntFactor.draw
- NodeTreeInterfaceSocketIntPercentage.
- NodeTreeInterfaceSocketIntUnsigned.dr
- NodeTreeInterfaceSocketMaterial.draw
- NodeTreeInterfaceSocketMatrix.draw
- NodeTreeInterfaceSocketMenu.draw
- NodeTreeInterfaceSocketObject.draw
- NodeTreeInterfaceSocketRotation.draw
- NodeTreeInterfaceSocketShader.draw
- NodeTreeInterfaceSocketString.draw
- NodeTreeInterfaceSocketStringFilePath
- NodeTreeInterfaceSocketTexture.draw
- NodeTreeInterfaceSocketVector.draw
- NodeTreeInterfaceSocketVectorAccelera
- NodeTreeInterfaceSocketVectorDirectio
- NodeTreeInterfaceSocketVectorEuler.dr
- NodeTreeInterfaceSocketVectorTranslat
- NodeTreeInterfaceSocketVectorVelocity
- NodeTreeInterfaceSocketVectorXYZ.draw
- Operator.cancel
- Operator.check
- Operator.description
- Operator.draw
- Operator.execute
- Operator.invoke
- Operator.modal
- Operator.poll
- Panel.draw
- Panel.draw_header
- Panel.draw_header_preset
- Panel.poll
- RenderEngine.draw
- RenderEngine.view_draw
- RenderEngine.view_update
- UIList.draw_filter
- UIList.draw_item
- UIList.filter_items
- XrSessionState.action_binding_create
- XrSessionState.action_create
- XrSessionState.action_set_create
- XrSessionState.action_state_get
- XrSessionState.active_action_set_set
- XrSessionState.controller_aim_locatio
- XrSessionState.controller_aim_rotatio

- NodeTreeInterfaceSocketFloatPercentage.draw
- NodeTreeInterfaceSocketFloatTime.draw
- NodeTreeInterfaceSocketFloatTimeAbsolute.draw
- NodeTreeInterfaceSocketFloatUnsigned.draw
- NodeTreeInterfaceSocketFloatWavelength.draw
- NodeTreeInterfaceSocketGeometry.draw
- NodeTreeInterfaceSocketImage.draw
- XrSessionState.controller_grip_locati
- XrSessionState.controller_grip_rotati
- XrSessionState.controller_pose_action
- XrSessionState.haptic_action_apply
- XrSessionState.haptic_action_stop
- XrSessionState.is_running
- XrSessionState.reset_to_base_pose