WindowManager(ID)

base classes — `bpy_struct`, `ID`

class bpy.types.**WindowManager(ID)**

Window manager data-block defining open windows and other user interface data

addon_filter

Filter add-ons by category

TYPE:

enum in [], default ‘’

addon_search

Filter by add-on name, author & category

TYPE:

string, default “”, (never None)

addon_support

Display support level

- `OFFICIAL` Official – Officially supported.
- `COMMUNITY` Community – Maintained by community developers.

TYPE:

enum set in {‘OFFICIAL’, ‘COMMUNITY’}, default {‘COMMUNITY’, ‘OFFICIAL’}

addon_tags

TYPE:

`BlExtDummyGroup`, (readonly)

asset_path_dummy

Full path to the Blender file containing the active asset

TYPE:

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

extension_search

Filter by extension name, author & category

TYPE:

string, default “”, (never None)

extension_show_panel_available

Only show installed extensions

TYPE:

boolean, default True

extension_show_panel_installed

Only show installed extensions

TYPE:

boolean, default True

extension_tags

TYPE:

`BlExtDummyGroup`, (readonly)

extension_type

Show extensions by type

- `ADDON` Add-ons – Only show add-ons.
- `THEME` Themes – Only show themes.

TYPE:

enum in ['ADDON', 'THEME'], default 'ADDON'

extensions_blocked

Number of installed extensions which are blocked

TYPE:

int in [-inf, inf], default 0

extensions_updates

Number of extensions with available update

TYPE:

int in [-inf, inf], default 0

is_interface_locked

If true, the interface is currently locked by a running job and data shouldn't be modified from application timers. Otherwise, the running job might conflict with the handler causing unexpected results or even crashes.

TYPE:

boolean, default False, (readonly)

keyconfigs

Registered key configurations

TYPE:

`KeyConfigurations bpy_prop_collection` of `KeyConfig`, (readonly)

operators

Operator registry

TYPE:

`bpy_prop_collection` of `Operator`, (readonly)

poselib_previous_action

TYPE:

`Action`

preset_name

Name for new preset

TYPE:

string, default "New Preset", (never None)

windows

Open windows

TYPE:

TYPE:

bpy_prop_collection of Window, (readonly)

xr_session_settings

TYPE:

XrSessionSettings, (readonly, never None)

xr_session_state

Runtime state information about the VR session

TYPE:

XrSessionState, (readonly)

clipboard

Clipboard text storage.

TYPE:

str

classmethod fileselect_add(operator)

Opens a file selector with an operator. The string properties 'filepath', 'filename', 'directory' and a 'files' collection are assigned when present in the operator. If 'filter_glob' property is present in the operator and it's not empty, it will be used as a file filter (example value: '*.zip;*.py;*.exe').

PARAMETERS:

operator (Operator) – Operator to call

classmethod modal_handler_add(operator)

Add a modal handler to the window manager, for the given modal operator (called by invoke() with self, just before returning {'RUNNING_MODAL'})

PARAMETERS:

operator (Operator) – Operator to call

RETURNS:

Whether adding the handler was successful

RETURN TYPE:

boolean

event_timer_add(time_step, *, window=None)

Add a timer to the given window, to generate periodic 'TIMER' events

PARAMETERS:

- **time_step** (*float in [0, inf]*) – Time Step, Interval in seconds between timer events
- **window** (Window, (optional)) – Window to attach the timer to, or None

RETURN TYPE:

Timer

event_timer_remove(timer)

event_timer_remove

classmethod gizmo_group_type_ensure(identifier)

Activate an existing widget group (when the persistent option isn't set)

PARAMETERS:

identifier (*string, (never None)*) – Gizmo group type name

classmethod gizmo_group_type_unlink_delayed(identifier)

Unlink a widget group (when the persistent option is set)

PARAMETERS:

identifier (*string, (never None)*) – Gizmo group type name

progress_begin(min, max)

Start progress report

PARAMETERS:

- **min** (*float in [-inf, inf]*) – min, any value in range [0,9999]
- **max** (*float in [-inf, inf]*) – max, any value in range [min+1,9998]

progress_update(value)

Update the progress feedback

PARAMETERS:

value (*float in [-inf, inf]*) – value, Any value between min and max as set in progress_begin()

progress_end()

Terminate progress report

classmethod invoke_props_popup(operator, event)

Operator popup invoke (show operator properties and execute it automatically on changes)

PARAMETERS:

- **operator** (`Operator`) – Operator to call
- **event** (`Event`) – Event

RETURNS:

result

RETURN TYPE:

enum set in Operator Return Items

classmethod invoke_props_dialog(operator, *, width=300, title='', confirm_text='', cancel_default=False, text_ctxt='', translate=True)

Operator dialog (non-autoexec popup) invoke (show operator properties and only execute it on click on OK button)

PARAMETERS:

- **operator** (`Operator`) – Operator to call
- **width** (*int in [0, inf], (optional)*) – Width of the popup
- **title** (*string, (optional, never None)*) – Title, Optional text to show as title of the popup
- **confirm_text** (*string, (optional, never None)*) – Confirm Text, Optional text to show instead to the default "OK" confirmation button text
- **cancel_default** (*boolean, (optional)*) – cancel_default
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

RETURNS:

result

RETURN TYPE:

enum set in Operator Return Items

classmethod invoke_search_popup(operator)

Operator search popup invoke which searches values of the operator's `bpy.types.Operator.bl_property` (which must be ar

EnumProperty), executing it on confirmation

PARAMETERS:

operator (`Operator`) – Operator to call

classmethod invoke_popup(operator, *, width=300)

Operator popup invoke (only shows operator's properties, without executing it)

PARAMETERS:

- **operator** (`Operator`) – Operator to call
- **width** (*int in [0, inf], (optional)*) – Width of the popup

RETURNS:

result

RETURN TYPE:

enum set in Operator Return Items

classmethod invoke_confirm(operator, event, *, title='', message='', confirm_text='', icon='NONE', text_ctxt='', translate=True)

Operator confirmation popup (only to let user confirm the execution, no operator properties shown)

PARAMETERS:

- **operator** (`Operator`) – Operator to call
- **event** (`Event`) – Event
- **title** (*string, (optional, never None)*) – Title, Optional text to show as title of the popup
- **message** (*string, (optional, never None)*) – Message, Optional first line of content text
- **confirm_text** (*string, (optional, never None)*) – Confirm Text, Optional text to show instead to the default "OK" confirmation button text
- **icon** (*enum in ['NONE', 'WARNING', 'QUESTION', 'ERROR', 'INFO'], (optional)*) – Icon, Optional icon displayed in the dialog
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

RETURNS:

result

RETURN TYPE:

enum set in Operator Return Items

classmethod popmenu_begin__internal(title, *, icon='NONE')

popmenu_begin__internal

PARAMETERS:

icon (enum in Icon Items, (optional)) – icon

RETURN TYPE:

`UIPopupMenu` , (never None)

classmethod popmenu_end__internal(menu)

popmenu_end__internal

classmethod popover_begin__internal(*, ui_units_x=0, from_active_button=False)

popover_begin__internal

PARAMETERS:

- **ui_units_x** (*int in [0, inf], (optional)*) – ui_units_x
- **from_active_button** (*boolean, (optional)*) – Use Button, Use the active button for positioning

RETURN TYPE:

UIPopover, (never None)

classmethod popover_end__internal(menu, *, keymap=None)

popover_end__internal

PARAMETERS:

keymap (KeyMap, (optional)) – Key Map, Active key map

classmethod piemenu_begin__internal(title, *, icon='NONE', event=None)

piemenu_begin__internal

PARAMETERS:

icon (enum in Icon Items, (optional)) – icon

RETURN TYPE:

UIPieMenu, (never None)

classmethod piemenu_end__internal(menu)

piemenu_end__internal

classmethod operator_properties_last(operator)

operator_properties_last

RETURN TYPE:

OperatorProperties, (never None)

print_undo_steps()

print_undo_steps

classmethod tag_script_reload()

Tag for refreshing the interface after scripts have been reloaded

popover(draw_func, *, ui_units_x=0, keymap=None, from_active_button=False)

popup_menu(draw_func, *, title='', icon='NONE')

Popup menus can be useful for creating menus without having to register menu classes.

Note that they will not block the scripts execution, so the caller can't wait for user input.

```
import bpy


def draw(self, context):
    self.layout.label(text="Hello World")


bpy.context.window_manager.popup_menu(draw, title="Greeting", icon='INFO')
```

popup_menu_pie(event, draw_func, *, title='', icon='NONE')

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_cursor_add(callback, args, space_type, region_type)

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

PARAMETERS:

- **callback** (*Callable[[Any, ..., tuple[int, int]], Any]*) – A function that will be called when the cursor is drawn. It gets the specified arguments as input with the mouse position (tuple) as last argument.
- **args** (*tuple[Any, ...]*) – Arguments that will be passed to the callback.
- **space_type** (*str*) – The space type the callback draws in; for example VIEW_3D. (bpy.types.Space.type)
- **region_type** (*str*) – The region type the callback draws in; usually WINDOW. (bpy.types.Region.type)

RETURNS:

Handler that can be removed later on.

RETURN TYPE:

object

classmethod draw_cursor_remove(handler)

Remove a draw cursor handler that was added previously.

PARAMETERS:

handler (*object*) – The draw cursor handler that should be removed.

Inherited Properties

- bpy_struct.id_data
- ID.name
- ID.name_full
- ID.id_type
- ID.session_uid
- ID.is_evaluated
- ID.original
- ID.users
- ID.use_fake_user
- ID.use_extra_user
- ID.is_embedded_data
- ID.is_missing
- ID.is_runtime_data
- ID.is_editable
- ID.tag
- ID.is_library_indirect
- ID.library
- ID.library_weak_reference
- ID.asset_data
- ID.override_library
- ID.preview

Inherited Functions

- bpy_struct.as_pointer
- bpy_struct.type_recast

- 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.values
- ID.rename
- ID.evaluated_get
- ID.copy
- ID.asset_mark
- ID.asset_clear
- ID.asset_generate_preview
- ID.override_create
- ID.override_hierarchy_create
- ID.user_clear
- ID.user_remap
- ID.make_local
- ID.user_of_id
- ID.animation_data_create
- ID.animation_data_clear
- ID.update_tag
- ID.preview_ensure
- ID.bl_rna_get_subclass
- ID.bl_rna_get_subclass_py

References

- BlendData.window_managers • Context.window_manager