UILayout(bpy_struct)

base class — bpy_struct

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

User interface layout in a panel or header

activate_init

When true, buttons defined in popups will be activated on first display (use so you can type into a field without having to click on it first)

TYPE:

boolean, default False

active

TYPE:

boolean, default False

active_default

When true, an operator button defined after this will be activated when pressing return(use with popup dialogs)

TYPE:

boolean, default False

alert

TYPE:

boolean, default False

alignment

TYPE:

enum in ['EXPAND', 'LEFT', 'CENTER', 'RIGHT'], default 'EXPAND'

direction

TYPE:

enum in ['HORIZONTAL', 'VERTICAL'], default 'HORIZONTAL', (readonly)

emboss

- NORMAL Regular – Draw standard button emboss style.
- NONE None – Draw only text and icons.
- PULLDOWN_MENU Pulldown Menu – Draw pulldown menu style.
- RADIAL_MENU Pie Menu – Draw radial menu style.
- NONE_OR_STATUS None or Status – Draw with no emboss unless the button has a coloring status like an animation state.

TYPE:

enum in ['NORMAL', 'NONE', 'PULLDOWN_MENU', 'RADIAL_MENU', 'NONE_OR_STATUS'], default 'NORMAL'

enabled

When false, this (sub)layout is grayed out

TYPE:

boolean, default False

operator_context

TYPE:

enum in Operator Context Items, default 'INVOKE_DEFAULT'

enum in Operator Context Items, default 'INVOKE_DEFAULT'

scale_x

Scale factor along the X for items in this (sub)layout

TYPE:

float in [0, inf], default 0.0

scale_y

Scale factor along the Y for items in this (sub)layout

TYPE:

float in [0, inf], default 0.0

ui_units_x

Fixed size along the X for items in this (sub)layout

TYPE:

float in [0, inf], default 0.0

ui_units_y

Fixed size along the Y for items in this (sub)layout

TYPE:

float in [0, inf], default 0.0

use_property_decorate

TYPE:

boolean, default False

use_property_split

TYPE:

boolean, default False

row(*, align=False, heading='', heading_ctxt='', translate=True)

Sub-layout. Items placed in this sublayout are placed next to each other in a row.

PARAMETERS:

- **align** (*boolean, (optional)*) – Align buttons to each other
- **heading** (*string, (optional, never None)*) – Heading, Label to insert into the layout for this sub-layout
- **heading_ctxt** (*string, (optional, never None)*) – Override automatic translation context of the given heading
- **translate** (*boolean, (optional)*) – Translate the given heading, when UI translation is enabled

RETURNS:

Sub-layout to put items in

RETURN TYPE:

`UILayout`

column(*, align=False, heading='', heading_ctxt='', translate=True)

Sub-layout. Items placed in this sublayout are placed under each other in a column.

PARAMETERS:

- **align** (*boolean, (optional)*) – Align buttons to each other
- **heading** (*string, (optional, never None)*) – Heading, Label to insert into the layout for this sub-layout
- **heading_ctxt** (*string, (optional, never None)*) – Override automatic translation context of the given heading
- **translate** (*boolean, (optional)*) – Translate the given heading, when UI translation is enabled

RETURNS:

Sub-layout to put items in

RETURN TYPE:

UILayout

panel(idname, *, default_closed=False)

Creates a collapsable panel. Whether it is open or closed is stored in the region using the given idname. This can only be used when the panel has the full width of the panel region available to it. So it can't be used in e.g. in a box or columns.

PARAMETERS:

- **idname** (*string, (never None)*) – Identifier of the panel
- **default_closed** (*boolean, (optional)*) – Open by Default, When true, the panel will be open the first time it is shown

RETURNS:

layout_header, Sub-layout to put items in, UILayout

layout_body, Sub-layout to put items in. Will be none if the panel is collapsed., UILayout

RETURN TYPE:

(UILayout, UILayout)

panel_prop(data, property)

Similar to *.panel(...)* but instead of storing whether it is open or closed in the region, it is stored in the provided boolean property. This should be used when multiple instances of the same panel can exist. For example one for every item in a collection property or list. This can only be used when the panel has the full width of the panel region available to it. So it can't be used in e.g. in a box or columns.

PARAMETERS:

- **data** (AnyType, (never None)) – Data from which to take the open-state property
- **property** (*string, (never None)*) – Identifier of the boolean property that determines whether the panel is open or closed

RETURNS:

layout_header, Sub-layout to put items in, UILayout

layout_body, Sub-layout to put items in. Will be none if the panel is collapsed., UILayout

RETURN TYPE:

(UILayout, UILayout)

column_flow(*, columns=0, align=False)

column_flow

PARAMETERS:

- **columns** (*int in [0, inf], (optional)*) – Number of columns, 0 is automatic
- **align** (*boolean, (optional)*) – Align buttons to each other

RETURNS:

Sub-layout to put items in

RETURN TYPE:

UILayout

grid_flow(*, row_major=False, columns=0, even_columns=False, even_rows=False, align=False)

grid_flow

PARAMETERS:

- **row_major** (*boolean, (optional)*) – Fill row by row, instead of column by column
- **columns** (*int in [-inf, inf], (optional)*) – Number of columns, positive are absolute fixed numbers, 0 is automatic, negative are automati multiple numbers along major axis (e.g. -2 will only produce 2, 4, 6 etc. columns for row major layout, and 2, 4, 6 etc. rows for column

major layout).

- **even_columns** (*boolean, (optional)*) – All columns will have the same width
- **even_rows** (*boolean, (optional)*) – All rows will have the same height
- **align** (*boolean, (optional)*) – Align buttons to each other

RETURNS:

Sub-layout to put items in

RETURN TYPE:

UILayout

box()

Sublayout (items placed in this sublayout are placed under each other in a column and are surrounded by a box)

RETURNS:

Sub-layout to put items in

RETURN TYPE:

UILayout

split(*, factor=0.0, align=False)

split

PARAMETERS:

- **factor** (*float in [0, 1], (optional)*) – Percentage, Percentage of width to split at (leave unset for automatic calculation)
- **align** (*boolean, (optional)*) – Align buttons to each other

RETURNS:

Sub-layout to put items in

RETURN TYPE:

UILayout

menu_pie()

Sublayout. Items placed in this sublayout are placed in a radial fashion around the menu center).

RETURNS:

Sub-layout to put items in

RETURN TYPE:

UILayout

classmethod icon(data)

Return the custom icon for this data, use it e.g. to get materials or texture icons.

PARAMETERS:

data (AnyType , (never None)) – Data from which to take the icon

RETURNS:

Icon identifier

RETURN TYPE:

int in [0, inf]

classmethod enum_item_name(data, property, identifier)

Return the UI name for this enum item

PARAMETERS:

- **data** (AnyType , (never None)) – Data from which to take property

- **property** (*string, (never None)*) – Identifier of property in data
- **identifier** (*string, (never None)*) – Identifier of the enum item

RETURNS:

UI name of the enum item

RETURN TYPE:

string, (never None)

classmethod enum_item_description(data, property, identifier)

Return the UI description for this enum item

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **identifier** (*string, (never None)*) – Identifier of the enum item

RETURNS:

UI description of the enum item

RETURN TYPE:

string, (never None)

classmethod enum_item_icon(data, property, identifier)

Return the icon for this enum item

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **identifier** (*string, (never None)*) – Identifier of the enum item

RETURNS:

Icon identifier

RETURN TYPE:

int in [0, inf]

prop(data, property, *, text='', text_ctxt='', translate=True, icon='NONE', placeholder='', expand=False, slider=False, toggle=-1, icon_only=False, event=False, full_event=False, emboss=True, index=-1, icon_value=0, invert_checkbox=False)

Item. Exposes an RNA item and places it into the layout.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **placeholder** (*string, (optional)*) – Hint describing the expected value when empty
- **expand** (*boolean, (optional)*) – Expand button to show more detail
- **slider** (*boolean, (optional)*) – Use slider widget for numeric values
- **toggle** (*int in [-1, 1], (optional)*) – Use toggle widget for boolean values, or a checkbox when disabled (the default is -1 which uses toggle only when an icon is displayed)
- **icon_only** (*boolean, (optional)*) – Draw only icons in buttons, no text
- **event** (*boolean, (optional)*) – Use button to input key events
- **full_event** (*boolean, (optional)*) – Use button to input full events including modifiers
- **emboss** (*boolean, (optional)*) – Draw the button itself, not just the icon/text. When false, corresponds to the 'NONE_OR_STATUS'

- **emboss** (*boolean, (optional)*) – Draw the button itself, not just the icon/text. When false, corresponds to the 'NONE_OR_STATUS' layout emboss type.
- **index** (*int in [-2, inf], (optional)*) – The index of this button, when set a single member of an array can be accessed, when set to -1 all array members are used
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item
- **invert_checkbox** (*boolean, (optional)*) – Draw checkbox value inverted

props_enum(data, property)

props_enum

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

prop_menu_enum(data, property, *, text='', text_ctxt='', translate=True, icon='NONE')

prop_menu_enum

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item

prop_with_popover(data, property, *, text='', text_ctxt='', translate=True, icon='NONE', icon_only=False, panel)

prop_with_popover

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **icon_only** (*boolean, (optional)*) – Draw only icons in tabs, no text
- **panel** (*string, (never None)*) – Identifier of the panel

prop_with_menu(data, property, *, text='', text_ctxt='', translate=True, icon='NONE', icon_only=False, menu)

prop_with_menu

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **icon_only** (*boolean, (optional)*) – Draw only icons in tabs, no text
- **menu** (*string, (never None)*) – Identifier of the menu

prop_tabs_enum(data, property, *, data_highlight=None, property_highlight='', icon_only=False)

prop_tabs_enum

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **data_highlight** (`AnyType` , (optional, never None)) – Data from which to take highlight property
- **property_highlight** (*string, (optional, never None)*) – Identifier of highlight property in data
- **icon_only** (*boolean, (optional)*) – Draw only icons in tabs, no text

prop_enum(data, property, value, *, text='', text_ctxt='', translate=True, icon='NONE')

prop_enum

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **value** (*string, (never None)*) – Enum property value
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item

prop_search(data, property, search_data, search_property, *, text='', text_ctxt='', translate=True, icon='NONE', results_are_suggestions=False)

prop_search

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **search_data** (`AnyType` , (never None)) – Data from which to take collection to search in
- **search_property** (*string, (never None)*) – Identifier of search collection property
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **results_are_suggestions** (*boolean, (optional)*) – Accept inputs that do not match any item

prop_decorator(data, property, *, index=-1)

prop_decorator

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **index** (*int in [-2, inf], (optional)*) – The index of this button, when set a single member of an array can be accessed, when set to -1 all array members are used

operator(operator, *, text='', text_ctxt='', translate=True, icon='NONE', emboss=True, depress=False, icon_value=0, search_weight=0.0)

Item. Places a button into the layout to call an Operator.

PARAMETERS:

- **operator** (*string, (never None)*) – Identifier of the operator
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item

- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **emboss** (*boolean, (optional)*) – Draw the button itself, not just the icon/text
- **depress** (*boolean, (optional)*) – Draw pressed in
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item
- **search_weight** (*float in [-inf, inf], (optional)*) – Search Weight, Influences the sorting when using menu-seach

RETURNS:

Operator properties to fill in

RETURN TYPE:

OperatorProperties

operator_menu_hold(operator, *, text='', text_ctxt='', translate=True, icon='NONE', emboss=True, depress=False, icon_value=0, menu)

Item. Places a button into the layout to call an Operator.

PARAMETERS:

- **operator** (*string, (never None)*) – Identifier of the operator
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **emboss** (*boolean, (optional)*) – Draw the button itself, not just the icon/text
- **depress** (*boolean, (optional)*) – Draw pressed in
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item
- **menu** (*string, (never None)*) – Identifier of the menu

RETURNS:

Operator properties to fill in

RETURN TYPE:

OperatorProperties

operator_enum(operator, property, *, icon_only=False)

operator_enum

PARAMETERS:

- **operator** (*string, (never None)*) – Identifier of the operator
- **property** (*string, (never None)*) – Identifier of property in operator
- **icon_only** (*boolean, (optional)*) – Draw only icons in buttons, no text

operator_menu_enum(operator, property, *, text='', text_ctxt='', translate=True, icon='NONE')

operator_menu_enum

PARAMETERS:

- **operator** (*string, (never None)*) – Identifier of the operator
- **property** (*string, (never None)*) – Identifier of property in operator
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item

RETURNS:

Operator properties to fill in

RETURN TYPE:

OperatorProperties

operatorProperties

label(*, text='', text_ctxt='', translate=True, icon='NONE', icon_value=0)

Item. Displays text and/or icon in the layout.

PARAMETERS:

- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item

menu(menu, *, text='', text_ctxt='', translate=True, icon='NONE', icon_value=0)

menu

PARAMETERS:

- **menu** (*string, (never None)*) – Identifier of the menu
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item

menu_contents(menu)

menu_contents

PARAMETERS:

- **menu** (*string, (never None)*) – Identifier of the menu

popover(panel, *, text='', text_ctxt='', translate=True, icon='NONE', icon_value=0)

popover

PARAMETERS:

- **panel** (*string, (never None)*) – Identifier of the panel
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item

popover_group(space_type, region_type, context, category)

popover_group

PARAMETERS:

- **space_type** (enum in Space Type Items) – Space Type
- **region_type** (enum in Region Type Items) – Region Type
- **context** (*string, (never None)*) – panel type context
- **category** (*string, (never None)*) – panel type category

separator(*, factor=1.0, type='AUTO')

Item. Inserts empty space into the layout between items.

PARAMETERS:

- **factor** (*float in [0, inf], (optional)*) – Percentage, Percentage of width to space (leave unset for default space)
- **type** (*enum in ['AUTO', 'SPACE', 'LINE'], (optional)*)

- **type** (*enum in ['AUTO', 'SPACE', 'LINE'], (optional)*) –

  Type, The type of the separator

  - `AUTO` Auto – Best guess at what type of separator is needed..
  - `SPACE` Empty space – Horizontal or Vertical empty space, depending on layout direction..
  - `LINE` Line – Horizontal or Vertical line, depending on layout direction..

separator_spacer()

Item. Inserts horizontal spacing empty space into the layout between items.

progress(*, text='', text_ctxt='', translate=True, factor=0.0, type='BAR')

Progress indicator

PARAMETERS:

- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **factor** (*float in [0, 1], (optional)*) – Factor, Amount of progress from 0.0f to 1.0f
- **type** (*enum in ['BAR', 'RING'], (optional)*) – Type, The type of progress indicator

context_pointer_set(name, data)

context_pointer_set

PARAMETERS:

- **name** (*string, (never None)*) – Name, Name of entry in the context
- **data** (`AnyType`) – Pointer to put in context

context_string_set(name, value)

context_string_set

PARAMETERS:

- **name** (*string, (never None)*) – Name, Name of entry in the context
- **value** (*string, (never None)*) – Value, String to put in context

template_header()

Inserts common Space header UI (editor type selector)

template_ID(data, property, *, new='', open='', unlink='', filter='ALL', live_icon=False, text='', text_ctxt='', translate=True)

template_ID

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **new** (*string, (optional, never None)*) – Operator identifier to create a new ID block
- **open** (*string, (optional, never None)*) – Operator identifier to open a file for creating a new ID block
- **unlink** (*string, (optional, never None)*) – Operator identifier to unlink the ID block
- **filter** (*enum in ['ALL', 'AVAILABLE'], (optional)*) – Optionally limit the items which can be selected
- **live_icon** (*boolean, (optional)*) – Show preview instead of fixed icon
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

template_ID_preview(data, property, *, new='', open='', unlink='', rows=0, cols=0, filter='ALL', hide_buttons=False)

template_ID_preview

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **new** (*string, (optional, never None)*) – Operator identifier to create a new ID block
- **open** (*string, (optional, never None)*) – Operator identifier to open a file for creating a new ID block
- **unlink** (*string, (optional, never None)*) – Operator identifier to unlink the ID block
- **rows** (*int in [0, inf], (optional)*) – Number of thumbnail preview rows to display
- **cols** (*int in [0, inf], (optional)*) – Number of thumbnail preview columns to display
- **filter** (*enum in ['ALL', 'AVAILABLE'], (optional)*) – Optionally limit the items which can be selected
- **hide_buttons** (*boolean, (optional)*) – Show only list, no buttons

template_any_ID(data, property, type_property, *, text='', text_ctxt='', translate=True)

template_any_ID

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **type_property** (*string, (never None)*) – Identifier of property in data giving the type of the ID-blocks to use
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

template_ID_tabs(data, property, *, new='', menu='', filter='ALL')

template_ID_tabs

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **new** (*string, (optional, never None)*) – Operator identifier to create a new ID block
- **menu** (*string, (optional, never None)*) – Context menu identifier
- **filter** (*enum in ['ALL', 'AVAILABLE'], (optional)*) – Optionally limit the items which can be selected

template_action(id, *, new='', unlink='', text='', text_ctxt='', translate=True)

template_action

PARAMETERS:

- **id** (`ID` , (never None)) – The data-block for which to select an Action
- **new** (*string, (optional, never None)*) – Operator identifier to create a new ID block
- **unlink** (*string, (optional, never None)*) – Operator identifier to unlink the ID block
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

template_search(data, property, search_data, search_property, *, new='', unlink='', text='', text_ctxt='', translate=True)

template_search

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **search_data** (`AnyType` , (never None)) – Data from which to take collection to search in
- **search_property** (*string, (never None)*) – Identifier of search collection property
- **new** (*string, (optional, never None)*) – Operator identifier to create a new item for the collection

- **unlink** (*string, (optional, never None)*) – Operator identifier to unlink or delete the active item from the collection
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

template_search_preview(data, property, search_data, search_property, *, new='', unlink='', text='', text_ctxt='', translate=True, rows=0, cols=0)

template_search_preview

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **search_data** (`AnyType` , (never None)) – Data from which to take collection to search in
- **search_property** (*string, (never None)*) – Identifier of search collection property
- **new** (*string, (optional, never None)*) – Operator identifier to create a new item for the collection
- **unlink** (*string, (optional, never None)*) – Operator identifier to unlink or delete the active item from the collection
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **rows** (*int in [0, inf], (optional)*) – Number of thumbnail preview rows to display
- **cols** (*int in [0, inf], (optional)*) – Number of thumbnail preview columns to display

template_path_builder(data, property, root, *, text='', text_ctxt='', translate=True)

template_path_builder

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **root** (`ID`) – ID-block from which path is evaluated from
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

template_modifiers()

Generates the UI layout for the modifier stack

template_collection_exporters()

Generates the UI layout for collection exporters

template_constraints(*, use_bone_constraints=True)

Generates the panels for the constraint stack

PARAMETERS:

use_bone_constraints (*boolean, (optional)*) – Add panels for bone constraints instead of object constraints

template_shaderfx()

Generates the panels for the shader effect stack

template_greasepencil_color(data, property, *, rows=0, cols=0, scale=1.0, filter='ALL')

template_greasepencil_color

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

- **property** (*string, (never None)*) – Identifier of property in data
- **rows** (*int in [0, inf], (optional)*) – Number of thumbnail preview rows to display
- **cols** (*int in [0, inf], (optional)*) – Number of thumbnail preview columns to display
- **scale** (*float in [0.1, 1.5], (optional)*) – Scale of the image thumbnails
- **filter** (*enum in ['ALL', 'AVAILABLE'], (optional)*) – Optionally limit the items which can be selected

template_constraint_header(data)

Generates the header for constraint panels

PARAMETERS:

data (`Constraint` , (never None)) – Constraint data

template_preview(id, *, show_buttons=True, parent=None, slot=None, preview_id='')

Item. A preview window for materials, textures, lights or worlds.

PARAMETERS:

- **id** (`ID`) – ID data-block
- **show_buttons** (*boolean, (optional)*) – Show preview buttons?
- **parent** (`ID` , (optional)) – ID data-block
- **slot** (`TextureSlot` , (optional)) – Texture slot
- **preview_id** (*string, (optional, never None)*) – Identifier of this preview widget, if not set the ID type will be used (i.e. all previews of materials without explicit ID will have the same size…).

template_curve_mapping(data, property, *, type='NONE', levels=False, brush=False, use_negative_slope=False, show_tone=False)

Item. A curve mapping widget used for e.g falloff curves for lights.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **type** (*enum in ['NONE', 'VECTOR', 'COLOR', 'HUE'], (optional)*) – Type, Type of curves to display
- **levels** (*boolean, (optional)*) – Show black/white levels
- **brush** (*boolean, (optional)*) – Show brush options
- **use_negative_slope** (*boolean, (optional)*) – Use a negative slope by default
- **show_tone** (*boolean, (optional)*) – Show tone options

template_curveprofile(data, property)

A profile path editor used for custom profiles

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_color_ramp(data, property, *, expand=False)

Item. A color ramp widget.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **expand** (*boolean, (optional)*) – Expand button to show more detail

template_icon(icon_value, *, scale=1.0)

Display a large icon

PARAMETERS:

- **icon_value** (*int in [0, inf]*) – Icon to display
- **scale** (*float in [1, 100], (optional)*) – Scale, Scale the icon size (by the button size)

template_icon_view(data, property, *, show_labels=False, scale=6.0, scale_popup=5.0)

Enum. Large widget showing Icon previews.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **show_labels** (*boolean, (optional)*) – Show enum label in preview buttons
- **scale** (*float in [1, 100], (optional)*) – UI Units, Scale the button icon size (by the button size)
- **scale_popup** (*float in [1, 100], (optional)*) – Scale, Scale the popup icon size (by the button size)

template_histogram(data, property)

Item. A histogramm widget to analyze imaga data.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_waveform(data, property)

Item. A waveform widget to analyze imaga data.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_vectorscope(data, property)

Item. A vectorscope widget to analyze imaga data.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_layers(data, property, used_layers_data, used_layers_property, active_layer)

template_layers

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **used_layers_data** (`AnyType`) – Data from which to take property
- **used_layers_property** (*string, (never None)*) – Identifier of property in data
- **active_layer** (*int in [0, inf]*) – Active Layer

template_color_picker(data, property, *, value_slider=False, lock=False, lock_luminosity=False, cubic=False)

Item. A color wheel widget to pick colors.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **value_slider** (*boolean, (optional)*) – Display the value slider to the right of the color wheel
- **lock** (*boolean, (optional)*) – Lock the color wheel display to value 1.0 regardless of actual color
- **lock_luminosity** (*boolean, (optional)*) – Keep the color at its original vector length
- **cubic** (*boolean, (optional)*) – Cubic saturation for picking values close to white

template_palette(data, property, *, color=False)

Item. A palette used to pick colors.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **color** (*boolean, (optional)*) – Display the colors as colors or values

template_image_layers(image, image_user)

template_image_layers

template_image(data, property, image_user, *, compact=False, multiview=False)

Item(s). User interface for selecting images and their source paths.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **compact** (*boolean, (optional)*) – Use more compact layout
- **multiview** (*boolean, (optional)*) – Expose Multi-View options

template_image_settings(image_settings, *, color_management=False)

User interface for setting image format options

PARAMETERS:

color_management (*boolean, (optional)*) – Show color management settings

template_image_stereo_3d(stereo_3d_format)

User interface for setting image stereo 3d options

template_image_views(image_settings)

User interface for setting image views output options

template_movieclip(data, property, *, compact=False)

Item(s). User interface for selecting movie clips and their source paths.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **compact** (*boolean, (optional)*) – Use more compact layout

template_track(data, property)

Item. A movie-track widget to preview tracking image.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_marker(data, property, clip_user, track, *, compact=False)

Item. A widget to control single marker settings.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data
- **compact** (*boolean, (optional)*) – Use more compact layout

template_movieclip_information(data, property, clip_user)

Item. Movie clip information data.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_list(listtype_name, list_id, dataptr, propname, active_dataptr, active_propname, *, item_dyntip_propname='', rows=5, maxrows=5, type='DEFAULT', columns=9, sort_reverse=False, sort_lock=False)

Item. A list widget to display data, e.g. vertexgroups.

PARAMETERS:

- **listtype_name** (*string, (never None)*) – Identifier of the list type to use
- **list_id** (*string, (never None)*) – Identifier of this list widget. Necessary to tell apart different list widgets. Mandatory when using default "UI_UL_list" class. If this not an empty string, the uilist gets a custom ID, otherwise it takes the name of the class used to define the uilist (for example, if the class name is "OBJECT_UL_vgroups", and list_id is not set by the script, then bl_idname = "OBJECT_UL_vgroups
- **dataptr** (`AnyType`) – Data from which to take the Collection property
- **propname** (*string, (never None)*) – Identifier of the Collection property in data
- **active_dataptr** (`AnyType` , (never None)) – Data from which to take the integer property, index of the active item
- **active_propname** (*string, (never None)*) – Identifier of the integer property in active_data, index of the active item
- **item_dyntip_propname** (*string, (optional, never None)*) – Identifier of a string property in items, to use as tooltip content
- **rows** (*int in [0, inf], (optional)*) – Default and minimum number of rows to display
- **maxrows** (*int in [0, inf], (optional)*) – Default maximum number of rows to display
- **type** (enum in Uilist Layout Type Items, (optional)) – Type, Type of layout to use
- **columns** (*int in [0, inf], (optional)*) – Number of items to display per row, for GRID layout
- **sort_reverse** (*boolean, (optional)*) – Display items in reverse order by default
- **sort_lock** (*boolean, (optional)*) – Lock display order to default value

template_running_jobs()

template_running_jobs

template_operator_search()

template_operator_search

template_menu_search()

template_menu_search

template_header_3D_mode()

template_edit_mode_selection()

Inserts common 3DView Edit modes header UI (selector for selection mode)

template_reports_banner()

template_reports_banner

template_input_status()

template_input_status

template_status_info()

template_status_info

template_node_link(ntree, node, socket)

template_node_link

template_node_view(ntree, node, socket)

template_node_view

template_node_asset_menu_items(*, catalog_path='')

template_node_asset_menu_items

template_modifier_asset_menu_items(*, catalog_path='')

template_modifier_asset_menu_items

template_node_operator_asset_menu_items(*, catalog_path='')

template_node_operator_asset_menu_items

template_node_operator_asset_root_items()

template_node_operator_asset_root_items

template_texture_user()

template_texture_user

template_keymap_item_properties(item)

template_keymap_item_properties

template_component_menu(data, property, *, name='')

Item. Display expanded property in a popup menu

PARAMETERS:

- **data** (`AnyType`) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_colorspace_settings(data, property)

Item. A widget to control input color space settings.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_colormanaged_view_settings(data, property)

Item. A widget to control color managed view settings.

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_node_socket(*, color=(0.0, 0.0, 0.0, 1.0))

Node Socket Icon

PARAMETERS:

color (*float array of 4 items in [0, 1], (optional)*) – Color

template_cache_file(data, property)

Item(s). User interface for selecting cache files and their source paths

PARAMETERS:

- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_cache_file_velocity(data, property)

Show cache files velocity properties

PARAMETERS:

- **data** (`AnyType`, (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_cache_file_procedural(data, property)

Show cache files render procedural properties

PARAMETERS:

- **data** (`AnyType`, (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_cache_file_time_settings(data, property)

Show cache files time settings

PARAMETERS:

- **data** (`AnyType`, (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_cache_file_layers(data, property)

Show cache files override layers properties

PARAMETERS:

- **data** (`AnyType`, (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_recent_files(*, rows=5)

Show list of recently saved .blend files

PARAMETERS:

rows (*int in [1, inf], (optional)*) – Maximum number of items to show

RETURNS:

Number of items drawn

RETURN TYPE:

int in [0, inf]

template_file_select_path(params)

Item. A text button to set the active file browser path.

template_event_from_keymap_item(item, *, text='', text_ctxt='', translate=True)

Display keymap item as icons/text

PARAMETERS:

- **item** (`KeyMapItem`, (never None)) – Item
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled

template_asset_view(list_id, asset_library_dataptr, asset_library_propname, assets_dataptr, assets_propname, active_dataptr, active_propname, *, filter_id_types={}, display_options={}, activate_operator='', drag_operator='')

Item. A scrollable list of assets in a grid view

PARAMETERS:

PARAMETERS:

- **list_id** (*string, (never None)*) – Identifier of this asset view. Necessary to tell apart different asset views and to identify an asset view re from a .blend
- **asset_library_dataptr** (`AnyType` , (never None)) – Data from which to take the active asset library property
- **asset_library_propname** (*string, (never None)*) – Identifier of the asset library property
- **assets_dataptr** (`AnyType` , (never None)) – Data from which to take the asset list property
- **assets_propname** (*string, (never None)*) – Identifier of the asset list property
- **active_dataptr** (`AnyType` , (never None)) – Data from which to take the integer property, index of the active item
- **active_propname** (*string, (never None)*) – Identifier of the integer property in active_data, index of the active item
- **filter_id_types** (*enum set in {}, (optional)*) – filter_id_types
- **display_options** (*enum set in {'NO_NAMES', 'NO_FILTER', 'NO_LIBRARY'}, (optional)*) – Displaying options for the asset view
  - `NO_NAMES` Do not display the name of each asset underneath preview images.
  - `NO_FILTER` Do not display buttons for filtering the available assets.
  - `NO_LIBRARY` Do not display buttons to choose or refresh an asset library.
- **activate_operator** (*string, (optional, never None)*) – Name of a custom operator to invoke when activating an item
- **drag_operator** (*string, (optional, never None)*) – Name of a custom operator to invoke when starting to drag an item. Never invoked together with the `active_operator` (if set), it's either the drag or the activate one

RETURNS:

activate_operator_properties, Operator properties to fill in for the custom activate operator passed to the template, `OperatorProperties`

drag_operator_properties, Operator properties to fill in for the custom drag operator passed to the template, `OperatorProperties`

RETURN TYPE:

(`OperatorProperties` , `OperatorProperties`)

template_light_linking_collection(context_layout, data, property)

Visualization of a content of a light linking collection

PARAMETERS:

- **context_layout** (`UILayout` , (never None)) – Layout to set active list element as context properties
- **data** (`AnyType` , (never None)) – Data from which to take property
- **property** (*string, (never None)*) – Identifier of property in data

template_bone_collection_tree()

Show bone collections tree

template_grease_pencil_layer_tree()

View of the active Grease Pencil layer tree

template_node_tree_interface(interface)

Show a node tree interface

PARAMETERS:

interface (`NodeTreeInterface` , (never None)) – Node Tree Interface, Interface of a node tree to display

template_node_inputs(node)

Show a node settings and input socket values

PARAMETERS:

node (`Node` , (never None)) – Node, Display inputs of this node

template_asset_shelf_popover(asset_shelf, *, name='', icon='NONE', icon_value=0)

Create a button to open an asset shelf in a popover

PARAMETERS:

- **asset_shelf** (*string, (never None)*) – Identifier of the asset shelf to display (`bl_idname`)
- **name** (*string, (optional)*) – Optional name to indicate the active asset
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **icon_value** (*int in [0, inf], (optional)*) – Icon Value, Override automatic icon of the item

template_popup_confirm(operator, *, text='', text_ctxt='', translate=True, icon='NONE', cancel_text='', cancel_default=False)

Add confirm & cancel buttons into a popup which will close the popup when pressed

PARAMETERS:

- **operator** (*string, (never None)*) – Identifier of the operator
- **text** (*string, (optional)*) – Override automatic text of the item
- **text_ctxt** (*string, (optional)*) – Override automatic translation context of the given text
- **translate** (*boolean, (optional)*) – Translate the given text, when UI translation is enabled
- **icon** (enum in Icon Items, (optional)) – Icon, Override automatic icon of the item
- **cancel_text** (*string, (optional, never None)*) – Optional text to use for the cancel, not shown when an empty string
- **cancel_default** (*boolean, (optional)*) – Cancel button by default

RETURNS:

Operator properties to fill in

RETURN TYPE:

`OperatorProperties`

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

introspect()

Return a dictionary containing a textual representation of the UI layout.

Inherited Properties

- `bpy_struct.id_data`

Inherited Functions

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
- Header.layout
- Menu.layout
- Node.draw_buttons
- Node.draw_buttons_ext
- NodeInternal.draw_buttons
- NodeInternal.draw_buttons_ext
- NodeSocket.draw
- NodeSocketStandard.draw
- NodeTreeInterfaceSocket.draw
- NodeTreeInterfaceSocketBool.draw
- NodeTreeInterfaceSocketCollection.draw
- NodeTreeInterfaceSocketColor.draw
- NodeTreeInterfaceSocketFloat.draw
- NodeTreeInterfaceSocketFloatAngle.draw
- NodeTreeInterfaceSocketFloatColorTemperature.draw
- NodeTreeInterfaceSocketFloatDistance.draw
- NodeTreeInterfaceSocketFloatFactor.draw
- NodeTreeInterfaceSocketFloatFrequency.draw
- NodeTreeInterfaceSocketFloatPercentage.draw
- NodeTreeInterfaceSocketFloatTime.draw
- NodeTreeInterfaceSocketFloatTimeAbsolute.draw
- NodeTreeInterfaceSocketFloatUnsigned.draw
- NodeTreeInterfaceSocketFloatWavelength.draw
- NodeTreeInterfaceSocketGeometry.draw
- NodeTreeInterfaceSocketImage.draw
- NodeTreeInterfaceSocketInt.draw
- NodeTreeInterfaceSocketIntFactor.draw
- NodeTreeInterfaceSocketIntPercentage.draw
- NodeTreeInterfaceSocketIntUnsigned.draw
- NodeTreeInterfaceSocketMaterial.draw
- NodeTreeInterfaceSocketMatrix.draw
- NodeTreeInterfaceSocketObject.draw
- NodeTreeInterfaceSocketRotation.draw
- NodeTreeInterfaceSocketShader.draw
- NodeTreeInterfaceSocketString.draw
- NodeTreeInterfaceSocketStringFilePath
- NodeTreeInterfaceSocketTexture.draw
- NodeTreeInterfaceSocketVector.draw
- NodeTreeInterfaceSocketVectorAccelera
- NodeTreeInterfaceSocketVectorDirectio
- NodeTreeInterfaceSocketVectorEuler.dr
- NodeTreeInterfaceSocketVectorTranslat
- NodeTreeInterfaceSocketVectorVelocity
- NodeTreeInterfaceSocketVectorXYZ.draw
- Operator.layout
- Panel.layout
- UILayout.box
- UILayout.column
- UILayout.column_flow
- UILayout.grid_flow
- UILayout.menu_pie
- UILayout.panel
- UILayout.panel
- UILayout.panel_prop
- UILayout.panel_prop
- UILayout.row
- UILayout.split
- UILayout.template_light_linking_colle
- UIList.draw_filter
- UIList.draw_item
- UIPieMenu.layout
- UIPopover.layout

- NodeTreeInterfaceSocketMenu.draw
- UIPopupMenu.layout