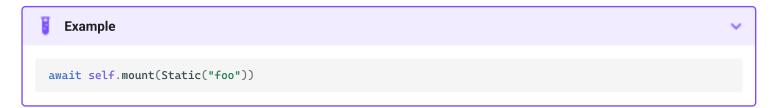
textual.widget

This module contains the Widget class, the base class for all widgets.

class AwaitMount

AwaitMount(parent, widgets)

An optional awaitable returned by mount and mount_all.



class BadWidgetName

Bases: Exception

Raised when widget class names do not satisfy the required restrictions.

class MountError

Bases: WidgetError

Error raised when there was a problem with the mount request.

class PseudoClasses

Bases: NamedTuple

Used for render/render_line based widgets that use caching. This structure can be used as a cache-key.

attr enabled

enabled

Is 'enabled' applied?

attr focus

instance-attribute

focus

Is 'focus' applied?

attr hover

instance-attribute

hover

Is 'hover' applied?

class Widget

```
Widget(
    *children,
    name=None,
    id=None,
    classes=None,
    disabled=False,
    markup=True
)
```

Bases: DOMNode

A Widget is the base class for Textual widgets.

See also static for starting point for your own widgets.

Parameters:

Name	Туре	Description	Default
*children	Widget	Child widgets.	O
name	str None	The name of the widget.	None
id	str None	The ID of the widget in the DOM.	None
classes	str None	The CSS classes for the widget.	None
disabled	bool	Whether the widget is disabled or not.	False

Name	Туре	Description	Default
markup	bool	Enable content markup?	True

attr ALLOW_MAXIMIZE

class-attribute

ALLOW_MAXIMIZE = None

Defines default logic to allow the widget to be maximized.

- None Use default behavior (Focusable widgets may be maximized)
- False Do not allow widget to be maximized
- True Allow widget to be maximized

attr ALLOW_SELECT

class-attribute

ALLOW_SELECT = True

Does this widget support automatic text selection? May be further refined with Widget.allow_select

attr BORDER_SUBTITLE

class-attribute

BORDER_SUBTITLE = ''

Initial value for border_subtitle attribute.

attr BORDER_TITLE

class-attribute

BORDER_TITLE = ''

Initial value for border_title attribute.

attr absolute_offset

instance-attribute

absolute_offset = None

Force an absolute offset for the widget (used by tooltips).

attr allow_horizontal_scroll

property

allow_horizontal_scroll

Check if horizontal scroll is permitted.

May be overridden if you want different logic regarding allowing scrolling.

attr allow_maximize

allow_maximize

Check if the widget may be maximized.

Returns:

Туре	Description
bool	True if the widget may be maximized, or False if it should not be maximized.

attr allow_select

property

property

allow_select

Check if this widget permits text selection.

Returns:

Туре	Description
bool	True if the widget supports text selection, otherwise False.

attr allow_vertical_scroll

property

allow_vertical_scroll

Check if vertical scroll is permitted.

May be overridden if you want different logic regarding allowing scrolling.

attr auto_links

instance-attribute class-attribute

auto_links = Reactive(True)

Widget will highlight links automatically.

attr border_subtitle

instance-attribute class-attribute

border_subtitle = _BorderTitle()

A title to show in the bottom border (if there is one).

attr border_title

instance-attribute class-attribute

```
border_title = _BorderTitle()
```

A title to show in the top border (if there is one).

attr can_focus

instance-attribute class-attribute

can_focus = False

Widget may receive focus.

attr can_focus_children

instance-attribute class-attribute

can_focus_children = True

Widget's children may receive focus.

attr container_scroll_offset

property

container_scroll_offset

The scroll offset the nearest container ancestor.

attr container_size

property

container_size

The size of the container (parent widget).

Returns:

Туре	Description
Size	Container size.

attr container_viewport

property

container_viewport

The viewport region (parent window).

Туре	Description
Region	The region that contains this widget.

attr content_offset property

content_offset

An offset from the Widget origin where the content begins.

Returns:

Туре	Description
Offset	Offset from widget's origin.

attr content_region

property

content_region

Gets an absolute region containing the content (minus padding and border).

Returns:

Туре	Description
Region	Screen region that contains a widget's content.

attr content_size

property

content_size

The size of the content area.

Returns:

Туре	Description
Size	Content area size.

attr disabled

instance-attribute class-attribute

disabled = Reactive(False)

Is the widget disabled? Disabled widgets can not be interacted with, and are typically styled to look dimmer.

attr dock_gutter

property

dock_gutter

Space allocated to docks in the parent.

Returns:

Туре	Description
Spacing	Space to be subtracted from scrollable area.

attr expand

instance-attribute class-attribute

expand = Reactive(False)

Rich renderable may expand beyond optimal size.

attr first_child property

first_child

Is this the first widget in its siblings?

attr first_of_type

property

first_of_type

Is this the first widget of its type in its siblings?

attr focusable

property

focusable

Can this widget currently be focused?

attr gutter

property

gutter

Spacing for padding / border / scrollbars.

Туре	Description
Spacing	Additional spacing around content area.

```
has_focus = Reactive(False, repaint=False)
   Does this widget have focus? Read only.
                                                                                                     property
attr has_focus_within
 has_focus_within
   Are any descendants focused?
                                                                          instance-attribute class-attribute
attr highlight_link_id
 highlight_link_id = Reactive('')
   The currently highlighted link id. Read only.
                                                                                                     property
attr horizontal_scrollbar
 horizontal_scrollbar
   The horizontal scrollbar.
         Note
     This will create a scrollbar if one doesn't exist.
   Returns:
                                                    Description
      Type
                                                    ScrollBar Widget.
      ScrollBar
                                                                          instance-attribute class-attribute
attr hover_style
 hover_style = Reactive(Style, repaint=False)
   The current hover style (style under the mouse cursor). Read only.
                                                                                                     property
attr is_anchored
 is_anchored
```

Is this widget anchored?

See anchor() for an explanation of anchoring.

```
property
attr is_container
 is_container
   Is this widget a container (contains other widgets)?
                                                                                                    property
attr is_disabled
 is_disabled
   Is the widget disabled either because disabled=True or an ancestor has disabled=True.
                                                                                                    property
attr is_even
 is_even
   Is this widget at an evenly numbered position within its siblings?
                                                                                                    property
attr is_horizontal_scroll_end
 is_horizontal_scroll_end
   Is the horizontal scroll position at the maximum?
attr is_horizontal_scrollbar_grabbed
                                                                                                    property
 is_horizontal_scrollbar_grabbed
   Is the user dragging the vertical scrollbar?
                                                                                                    property
attr is_in_maximized_view
 is_in_maximized_view
   Is this widget, or a parent maximized?
                                                                                                    property
attr is_maximized
 is_maximized
   Is this widget maximized?
                                                                                                    property
attr is_mounted
 is_mounted
   Check if this widget is mounted.
```

```
is_mouse_over
   Is the mouse currently over this widget?
   Note this will be True if the mouse pointer is within the widget's region, even if the mouse pointer is not directly
   over the widget (there could be another widget between the mouse pointer and self).
                                                                                                        property
attr is_odd
 is_odd
   Is this widget at an oddly numbered position within its siblings?
                                                                                                        property
attr is_on_screen
 is_on_screen
   Check if the node was displayed in the last screen update.
                                                                                                        property
attr is_scrollable
 is_scrollable
   Can this widget be scrolled?
                                                                                                        property
attr is_scrolling
 is_scrolling
   Is this widget currently scrolling?
                                                                                                        property
attr is_vertical_scroll_end
 is_vertical_scroll_end
   Is the vertical scroll position at the maximum?
                                                                                                        property
attr is_vertical_scrollbar_grabbed
 is_vertical_scrollbar_grabbed
   Is the user dragging the vertical scrollbar?
                                                                                                        property
attr last_child
```

attr is_mouse_over

property

last_child

Is this the last widget in its siblings?

attr last_of_type

property

last_of_type

Is this the last widget of its type in its siblings?

attr layer

property

layer

Get the name of this widgets layer.

Returns:

Туре	Description
str	Name of layer.

attr layers

property

layers

Layers of from parent.

Returns:

Туре	Description
<pre>tuple[str,]</pre>	Tuple of layer names.

attr layout

property

layout

Get the layout object if set in styles, or a default layout.

Туре	Description
Layout	A layout object.

attr link_style property

link_style

Style of links.

Returns:

Туре	Description	
Style	Rich style.	

attr link_style_hover

property

link_style_hover

Style of links underneath the mouse cursor.

Returns:

Туре	Description
Style	Rich Style.

attr loading

instance-attribute class-attribute

loading = Reactive(False)

If set to True this widget will temporarily be replaced with a loading indicator.

attr lock instance-attribute

lock = RLock()

asyncio lock to be used to synchronize the state of the widget.

Two different tasks might call methods on a widget at the same time, which might result in a race condition. This can be fixed by adding async with widget.lock: around the method calls.

attr max_scroll_x

property

max_scroll_x

The maximum value of scroll_x.

attr max_scroll_y

property

```
max_scroll_y
```

The maximum value of scroll_y.

attr mouse_hover

instance-attribute class-attribute

mouse_hover = Reactive(False, repaint=False)

Is the mouse over this widget? Read only.

attr offset

writable property

offset

Widget offset from origin.

Returns:

Туре	Description
Offset	Relative offset.

attr opacity

property

opacity

Total opacity of widget.

attr outer_size

property

outer_size

The size of the widget (including padding and border).

Returns:

Туре	Description
Size	Outer size.

attr region

property

region

The region occupied by this widget, relative to the Screen.

Raises:

Туре	Description
NoScreen	If there is no screen.
NoWidget	If the widget is not on the screen.

Returns:

Туре	Description
Region	Region within screen occupied by widget.

attr scroll_offset property

scroll_offset

Get the current scroll offset.

Returns:

Туре	Description
Offset	Offset a container has been scrolled by.

attr scroll_target_x

instance-attribute class-attribute

```
scroll_target_x = Reactive(0.0, repaint=False)
```

Scroll target destination, X coord.

attr scroll_target_y

instance-attribute class-attribute

```
scroll_target_y = Reactive(0.0, repaint=False)
```

Scroll target destination, Y coord.

attr scroll_x

instance-attribute class-attribute

```
scroll_x = Reactive(0.0, repaint=False, layout=False)
```

The scroll position on the X axis.

attr scroll_y

instance-attribute class-attribute

```
scroll_y = Reactive(0.0, repaint=False, layout=False)
```

The scroll position on the Y axis.

attr scrollable_content_region

property

scrollable_content_region

Gets an absolute region containing the scrollable content (minus padding, border, and scrollbars).

Returns:

Туре	Description
Region	Screen region that contains a widget's content.

attr scrollable_size

property

scrollable_size

The size of the scrollable content.

Returns:

Туре	Description
Size	Scrollable content size.

attr scrollbar_corner

property

scrollbar_corner

The scrollbar corner.



Note

This will create a scrollbar corner if one doesn't exist.

Туре	Description
ScrollBarCorner	ScrollBarCorner Widget.

scrollbar_gutter

Spacing required to fit scrollbar(s).

Returns:

Туре	Description
Spacing	Scrollbar gutter spacing.

attr scrollbar_size_horizontal

property

scrollbar_size_horizontal

Get the height used by the horizontal scrollbar.

Returns:

Туре	Description
int	Number of rows in the horizontal scrollbar.

attr scrollbar_size_vertical

property

scrollbar_size_vertical

Get the width used by the vertical scrollbar.

Returns:

Туре	Description
int	Number of columns in the vertical scrollbar.

attr scrollbars_enabled

property

scrollbars_enabled

A tuple of booleans that indicate if scrollbars are enabled.

Type	Description

Туре	Description
<pre>tuple[bool, bool]</pre>	A tuple of (,)

attr scrollbars_space

property

scrollbars_space

The number of cells occupied by scrollbars for width and height

attr select_container

property

select_container

The widget's container used when selecting text..

Returns:

Туре	Description
Widget	A widget which contains this widget.

attr show_horizontal_scrollbar

instance-attribute class-attribute

show_horizontal_scrollbar = Reactive(False, layout=True)

Show a horizontal scrollbar?

attr show_vertical_scrollbar

instance-attribute class-attribute

show_vertical_scrollbar = Reactive(False, layout=True)

Show a vertical scrollbar?

attr shrink

instance-attribute class-attribute

shrink = Reactive(True)

Rich renderable may shrink below optimal size.

attr siblings

property

siblings

Get the widget's siblings (self is removed from the return list).

Returns: Type **Description** A list of siblings. list[Widget] property attr size size The size of the content area. **Returns: Description** Type Size Content area size. attr text_selection property text_selection Text selection information, or None if no text is selected in this widget. writable property attr tooltip tooltip Tooltip for the widget, or None for no tooltip. property attr vertical_scrollbar vertical_scrollbar The vertical scrollbar (create if necessary). Note This will create a scrollbar if one doesn't exist. **Returns:**

Description

ScrollBar Widget.

Type

ScrollBar

attr virtual_region property

virtual_region

The widget region relative to its container (which may not be visible, depending on scroll offset).

Returns:

Туре	Description
Region	The virtual region.

attr virtual_region_with_margin

property

virtual_region_with_margin

The widget region relative to its container (*including margin*), which may not be visible, depending on the scroll offset.

Returns:

Туре	Description
Region	The virtual region of the Widget, inclusive of its margin.

attr virtual_size

instance-attribute class-attribute

virtual_size = Reactive(Size(0, 0), layout=True)

The virtual (scrollable) [size][textual.geometry.Size] of the widget.

attr visible_siblings

property

visible_siblings

A list of siblings which will be shown.

Returns:

Туре	Description
list[Widget]	List of siblings.

attr window_region

property

The region within the scrollable area that is currently visible.

Returns:

Туре	Description
Region	New region.

meth allow_focus

allow_focus()

Check if the widget is permitted to focus.

The base class returns can_focus. This method may be overridden if additional logic is required.

Returns:

Туре	Description
bool	True if the widget may be focused, or False if it may not be focused.

meth allow_focus_children

allow_focus_children()

Check if a widget's children may be focused.

The base class returns <code>can_focus_children</code> . This method may be overridden if additional logic is required.

Returns:

Туре	Description
bool	True if the widget's children may be focused, or False if the widget's children may not be focused.

meth anchor

anchor(anchor=True)

Anchor a scrollable widget.

An anchored widget will stay scrolled the bottom when new content is added, until the user moves the scroll position.

Parameters:

Name	Туре	Description	Default
anchor	bool	Anchor the widget if $True$, clear the anchor if $False$.	True

meth animate

```
animate(
   attribute,
   value,
   *,
   final_value=...,
   duration=None,
   speed=None,
   delay=0.0,
   easing=DEFAULT_EASING,
   on_complete=None,
   level="full"
)
```

Animate an attribute.

Parameters:

Name	Туре	Description	Default
attribute	str	Name of the attribute to animate.	required
value	float Animatable	The value to animate to.	required
final_value	object	The final value of the animation. Defaults to value if not set.	
duration	float None	The duration (in seconds) of the animation.	None
speed	float None	The speed of the animation.	None
delay	float	A delay (in seconds) before the animation starts.	0.0
easing	EasingFunction str	An easing method.	DEFAULT_EASING
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'full'

meth batch async

batch()

Async context manager that combines widget locking and update batching.

Use this async context manager whenever you want to acquire the widget lock and batch app updates at the same time.

```
async with container.batch():
    await container.remove_children(Button)
    await container.mount(Label("All buttons are gone."))
```

meth begin_capture_print

```
begin_capture_print(stdout=True, stderr=True)
```

Capture text from print statements (or writes to stdout / stderr).

If printing is captured, the widget will be sent an events.Print message.

Call end_capture_print to disable print capture.

Parameters:

Name	Туре	Description	Default
stdout	bool	Whether to capture stdout.	True
stderr	bool	Whether to capture stderr.	True

meth blur

blur()

Blur (un-focus) the widget.

Focus will be moved to the next available widget in the focus chain.

Туре	Description
Self	The Widget instance.

meth can_view_entire

can_view_entire(widget)

Check if a given widget is fully within the current view (scrollable area).

Note: This doesn't necessarily equate to a widget being visible. There are other reasons why a widget may not be visible.

Parameters:

Name	Туре	Description	Default
widget	Widget	A widget that is a descendant of self.	required

Returns:

Туре	Description
bool	True if the entire widget is in view, False if it is partially visible or not in view.

meth can_view_partial

can_view_partial(widget)

Check if a given widget at least partially visible within the current view (scrollable area).

Parameters:

Name	Туре	Description	Default
widget	Widget	A widget that is a descendant of self.	required

Returns:

Туре	Description
bool	True if any part of the widget is visible, False if it is outside of the viewable area.

meth capture_mouse

capture_mouse(capture=True)

Capture (or release) the mouse.

When captured, mouse events will go to this widget even when the pointer is not directly over the widget.

Parameters:

Name	Туре	Description	Default
capture	bool	True to capture or False to release.	True

meth check_message_enabled

check_message_enabled(message)

Check if a given message is enabled (allowed to be sent).

Parameters:

Name	Туре	Description	Default
message	Message	A message object	required

Returns:

Туре	Description
bool	True if the message will be sent, or False if it is disabled.

meth clear_cached_dimensions

clear_cached_dimensions()

Clear cached results of get_content_width and get_content_height.

Call if the widget's renderable changes size after the widget has been created.



Note

This is not required if you are extending Static .

meth compose

compose()

Called by Textual to create child widgets.

This method is called when a widget is mounted or by setting recompose=True when calling refresh().

Note that you don't typically need to explicitly call this method.

```
def compose(self) -> ComposeResult:
    yield Header()
    yield Label("Press the button below:")
    yield Button()
    yield Footer()
```

meth compose_add_child

```
compose_add_child(widget)
```

Add a node to children.

This is used by the compose process when it adds children. There is no need to use it directly, but you may want to override it in a subclass if you want children to be attached to a different node.

Parameters:

Name	Туре	Description	Default
widget	Widget	A Widget to add.	required

meth end_capture_print

```
end_capture_print()
```

End print capture (set with begin_capture_print).

meth focus

```
focus(scroll_visible=True)
```

Give focus to this widget.

Parameters:

Name	Туре	Description	Default
scroll_visible	bool	Scroll parent to make this widget visible.	True

Туре	Description
Self	The Widget instance.

meth get_child_by_id

```
get_child_by_id(id: str) -> Widget
```

```
get_child_by_id(
    id: str, expect_type: type[ExpectType]
) -> ExpectType
```

```
get_child_by_id(id, expect_type=None)
```

Return the first child (immediate descendent) of this node with the given ID.

Parameters:

Name	Туре	Description	Default
id	str	The ID of the child.	required
expect_type	<pre>type[ExpectType] None</pre>	Require the object be of the supplied type, or None for any type.	None

Returns:

Туре	Description
ExpectType Widget	The first child of this node with the ID.

Raises:

Туре	Description
NoMatches	if no children could be found for this ID
WrongType	if the wrong type was found.

meth get_child_by_type

```
get_child_by_type(expect_type)
```

Get the first immediate child of a given type.

Only returns exact matches, and so will not match subclasses of the given type.

Parameters:

Name	Туре	Description	Default

Name	Туре	Description	Default
expect_type	<pre>type[ExpectType]</pre>	The type of the child to search for.	required

Raises:

Туре	Description
NoMatches	If no matching child is found.

Returns:

Туре	Description
ExpectType	The first immediate child widget with the expected type.

meth get_component_rich_style

get_component_rich_style(*names, partial=False)

Get a *Rich* style for a component.

Parameters:

Name	Туре	Description	Default
names	str	Names of components.	C)
partial	bool	Return a partial style (not combined with parent).	False

Returns:

Туре	Description
Style	A Rich style object.

meth get_content_height

get_content_height(container, viewport, width)

Called by Textual to get the height of the content area. May be overridden in a subclass.

Parameters:

Name	Туре	Description	Default
container	Size	Size of the container (immediate parent) widget.	required
viewport	Size	Size of the viewport.	required
width	int	Width of renderable.	required

Returns:

Туре	Description
int	The height of the content.

meth get_content_width

get_content_width(container, viewport)

Called by textual to get the width of the content area. May be overridden in a subclass.

Parameters:

Name	Туре	Description	Default
container	Size	Size of the container (immediate parent) widget.	required
viewport	Size	Size of the viewport.	required

Returns:

Туре	Description
int	The optimal width of the content.

meth get_loading_widget

get_loading_widget()

Get a widget to display a loading indicator.

The default implementation will defer to App.get_loading_widget.

Type Description	

Туре	Description
Widget	A widget in place of this widget to indicate a loading.

meth get_pseudo_class_state

get_pseudo_class_state()

Get an object describing whether each pseudo class is present on this object or not.

Returns:

Туре	Description
PseudoClasses	A PseudoClasses object describing the pseudo classes that are present.

meth get_selection

get_selection(selection)

Get the text under the selection.

Args:

selection: Selection information.

Returns:

Tuple of extracted text and ending (typically "

meth get_style_at

get_style_at(x, y)

Get the Rich style in a widget at a given relative offset.

Parameters:

Name	Туре	Description	Default
х	int	X coordinate relative to the widget.	required
у	int	Y coordinate relative to the widget.	required

[&]quot; or " "), or None if no text could be extracted.

Туре	Description
Style	A rich Style object.

meth get_visual_style

```
get_visual_style(*component_classes, partial=False)
```

Get the visual style for the widget, including any component styles.

Parameters:

Name	Туре	Description	Default
component_classes	str	Optional component styles.	O
partial	bool	Return a partial style (not combined with parent).	False

Returns:

Туре	Description
Style	A Visual style instance.

meth get_widget_by_id

```
get_widget_by_id(id: str) -> Widget
```

```
get_widget_by_id(
    id: str, expect_type: type[ExpectType]
) -> ExpectType
```

```
get_widget_by_id(id, expect_type=None)
```

Return the first descendant widget with the given ID.

Performs a depth-first search rooted at this widget.

Parameters:

Name	Туре	Description	Default
id	str	The ID to search for in the subtree.	required
expect_type	<pre>type[ExpectType] None</pre>	Require the object be of the supplied type, or None for any type.	None

Returns:

Туре	Description
ExpectType Widget	The first descendant encountered with this ID.

Raises:

Туре	Description
NoMatches	if no children could be found for this ID.
WrongType	if the wrong type was found.

meth mount

mount(*widgets, before=None, after=None)

Mount widgets below this widget (making this widget a container).

Parameters:

Name	Туре	Description	Default
*widgets	Widget	The widget(s) to mount.	()
before	int str Widget None	Optional location to mount before. An int is the index of the child to mount before, a str is a query_one query to find the widget to mount before.	None
after	int str Widget None	Optional location to mount after. An int is the index of the child to mount after, a str is a query_one query to find the widget to mount after.	None

Returns:

Туре	Description
AwaitMount	An awaitable object that waits for widgets to be mounted.

Raises:

Туре	Description
MountError	If there is a problem with the mount request.



Only one of before or after can be provided. If both are provided a MountError will be raised.

meth mount_all

mount_all(widgets, *, before=None, after=None)

Mount widgets from an iterable.

Parameters:

Name	Туре	Description	Default
widgets	<pre>Iterable[Widget]</pre>	An iterable of widgets.	required
before	<pre>int str Widget None</pre>	Optional location to mount before. An int is the index of the child to mount before, a str is a query_one query to find the widget to mount before.	None
after	int str Widget None	Optional location to mount after. An int is the index of the child to mount after, a str is a query_one query to find the widget to mount after.	None

Returns:

Туре	Description
AwaitMount	An awaitable object that waits for widgets to be mounted.

Raises:

Туре	Description
MountError	If there is a problem with the mount request.



Only one of before or after can be provided. If both are provided a MountError will be raised.

Called by Textual to mount widgets after compose.

There is generally no need to implement this method in your application. See Lazy for a class which uses this method to implement *lazy* mounting.

Parameters:

Name	Туре	Description	Default
widgets	list[Widget]	A list of child widgets.	required

meth move_child

```
move_child(
    child: int | Widget,
    *,
    before: int | Widget,
    after: None = None
) -> None
```

```
move_child(
    child: int | Widget,
    *,
    after: int | Widget,
    before: None = None
) -> None
```

```
move_child(child, *, before=None, after=None)
```

Move a child widget within its parent's list of children.

Parameters:

Name	Туре	Description	Default
child	int Widget	The child widget to move.	required
before	int Widget None	Child widget or location index to move before.	None
after	int Widget None	Child widget or location index to move after.	None

Raises:

Туре	Description
WidgetError	If there is a problem with the child or target.



Only one of before or after can be provided. If neither or both are provided a WidgetError will be raised.

meth notify

```
notify(
    message,
    title="",
    severity="information",
    timeout=None,
    markup=True
)
```

Create a notification.



This method is thread-safe.

Parameters:

Name	Туре	Description	Default
message	str	The message for the notification.	required
title	str	The title for the notification.	11
severity	SeverityLevel	The severity of the notification.	'information'
timeout	float None	The timeout (in seconds) for the notification, or None for default.	None
markup	bool	Render the message as content markup?	True

See App.notify for the full documentation for this method.

meth on_prune

async

on_prune(event)

Close message loop when asked to prune.

meth post_message

```
post_message(message)
```

Post a message to this widget.

Parameters:

Name	Туре	Description	Default
message	Message	Message to post.	required

Returns:

Туре	Description
bool	True if the message was posted, False if this widget was closed / closing.

meth post_render

post_render(renderable, base_style)

Applies style attributes to the default renderable.

This method is called by Textual itself. It is unlikely you will need to call or implement this method.

Returns:

Туре	Description
ConsoleRenderable	A new renderable.

meth pre_layout

pre_layout(layout)

This method id called prior to a layout operation.

Implement this method if you want to make updates that should impact the layout.

Parameters:

Name	Туре	Description	Default
layout	Layout	The Layout instance that will be used to arrange this widget's children.	required

meth preflight_checks

preflight_checks()

Called in debug mode to do preflight checks.

This is used by Textual to log some common errors, but you could implement this in custom widgets to perform additional checks.

meth recompose async

```
recompose()
```

Recompose the widget.

Recomposing will remove children and call self.compose again to remount.

meth refresh

```
refresh(
   *regions, repaint=True, layout=False, recompose=False
)
```

Initiate a refresh of the widget.

This method sets an internal flag to perform a refresh, which will be done on the next idle event. Only one refresh will be done even if this method is called multiple times.

By default this method will cause the content of the widget to refresh, but not change its size. You can also set layout=True to perform a layout.



Warning

It is rarely necessary to call this method explicitly. Updating styles or reactive attributes will do this automatically.

Parameters:

Name	Туре	Description	Default
*regions	Region	Additional screen regions to mark as dirty.	()
repaint	bool	Repaint the widget (will call render() again).	True
layout	bool	Also layout widgets in the view.	False
recompose	bool	Re-compose the widget (will remove and re-mount children).	False

Туре	Description
Self	The Widget instance.

meth release_anchor

```
release_anchor()
```

Release the anchor.

If a widget is anchored, releasing the anchor will allow the user to scroll as normal.

meth release_mouse

```
release_mouse()
```

Release the mouse.

Mouse events will only be sent when the mouse is over the widget.

meth remove

remove()

Remove the Widget from the DOM (effectively deleting it).

Returns:

Туре	Description
AwaitRemove	An awaitable object that waits for the widget to be removed.

meth remove_children

```
remove_children(selector='*')
```

Remove the immediate children of this Widget from the DOM.

Parameters:

Name	Туре	Description	Default
selector	<pre>str type[QueryType] Iterable[Widget]</pre>	A CSS selector or iterable of widgets to remove.	**

Returns:

Туре	Description
AwaitRemove	An awaitable object that waits for the direct children to be removed.

meth render

render()

Get content for the widget.

Implement this method in a subclass for custom widgets.

This method should return markup, a Content object, or a Rich renderable.

```
from textual.app import RenderResult
from textual.widget import Widget

class CustomWidget(Widget):
    def render(self) -> RenderResult:
        return "Welcome to [bold red]Textual[/]!"
```

Returns:

Туре	Description
RenderResult	A string or object to render as the widget's content.

meth render_line

render_line(y)

Render a line of content.

Parameters:

Name	Туре	Description	Default
у	int	Y Coordinate of line.	required

Returns:

Туре	Description
Strip	A rendered line.

meth render_lines

render_lines(crop)

Render the widget into lines.

Parameters:

Name	Туре	Description	Default
crop	Region	Region within visible area to render.	required

Returns:

Туре	Description
list[Strip]	A list of list of segments.

meth render_str

render_str(text_content: str) -> Content

render_str(text_content: Content) -> Content

render_str(text_content)

Convert str into a Content instance.

If you pass in an existing Content instance it will be returned unaltered.

Parameters:

Name	Туре	Description	Default
text_content	str Content	Content or str.	required

Returns:

Туре	Description
Content	Content object.

meth run_action async

run_action(action)

Perform a given action, with this widget as the default namespace.

Name	Туре	Description	Default
action	str	Action encoded as a string.	required

meth scroll_down

```
scroll_down(
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False
)
```

Scroll one line down.

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

```
scroll_end(
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False,
    x_axis=True,
    y_axis=True
)
```

Scroll to the end of the container.

Parameters:

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False
x_axis	bool	Allow scrolling on X axis?	True
y_axis	bool	Allow scrolling on Y axis?	True

meth scroll_home

```
*,
animate=True,
speed=None,
duration=None,
easing=None,
force=False,
on_complete=None,
level="basic",
immediate=False,
x_axis=True,
y_axis=True
```

Scroll to home position.

Parameters:

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False
x_axis	bool	Allow scrolling on X axis?	True
y_axis	bool	Allow scrolling on Y axis?	True

meth scroll_left

```
scroll_left(
    *.
```

```
animate=True,
speed=None,
duration=None,
easing=None,
force=False,
on_complete=None,
level="basic",
immediate=False)
```

Scroll one cell left.

Parameters:

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

meth scroll_page_down

Scroll one page down.

Parameters:

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'

meth scroll_page_left

```
scroll_page_left(
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic"
)
```

Scroll one page left.

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; or None to use duration.	None

Name	Туре	Description	Default
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'

meth scroll_page_right

Scroll one page right.

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False

Name	Туре	Description	Default
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'

meth scroll_page_up

```
scroll_page_up(
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic"
)
```

Scroll one page up.

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'

```
scroll_relative(
    x=None,
    y=None,
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False
)
```

Scroll relative to current position.

Name	Туре	Description	Default
х	float None	X distance (columns) to scroll, or None for no change.	None
у	float None	Y distance (rows) to scroll, or None for no change.	None
animate	bool	Animate to new scroll position.	True
speed	float None	Speed of scroll if animate is True. Or None to use duration.	None
duration	float None	Duration of animation, if animate is $\ensuremath{\mathtt{True}}$ and speed is $\ensuremath{\mathtt{None}}$.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

```
scroll_right(
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False
)
```

Scroll one cell right.

Parameters:

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

meth scroll_to

```
scroll_to(
    x=None,
    y=None,
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
```

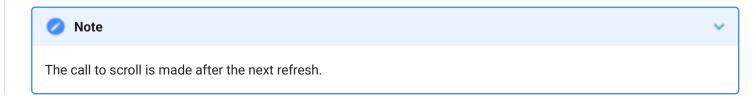
```
on_complete=None,
level="basic",
immediate=False,
release_anchor=True
```

Scroll to a given (absolute) coordinate, optionally animating.

Parameters:

)

Name	Туре	Description	Default
х	float None	X coordinate (column) to scroll to, or None for no change.	None
У	float None	Y coordinate (row) to scroll to, or None for no change.	None
animate	bool	Animate to new scroll position.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False
release_anchor	bool	If True Call release_anchor.	True



meth scroll_to_center

```
scroll_to_center(
    widget,
    animate=True,
    *,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    origin_visible=True,
    on_complete=None,
    level="basic",
    immediate=False
)
```

Scroll this widget to the center of self.

The center of the widget will be scrolled to the center of the container.

Name	Туре	Description	Default
widget	Widget	The widget to scroll to the center of self.	required
animate	bool	Whether to animate the scroll.	True
speed	float None	Speed of scroll if animate is True; or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
origin_visible	bool	Ensure that the top left corner of the widget remains visible after the scroll.	True
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

meth scroll_to_region

```
scroll_to_region(
   region,
    spacing=None,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    center=False,
    top=False,
    origin_visible=True,
    force=False,
    on_complete=None,
    level="basic",
    x_axis=True,
   y_axis=True,
    immediate=False
```

Scrolls a given region into view, if required.

This method will scroll the least distance required to move region fully within the scrollable area.

Name	Туре	Description	Default
region	Region	A region that should be visible.	required
spacing	Spacing None	Optional spacing around the region.	None
animate	bool	True to animate, or False to jump.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
top	bool	Scroll region to top of container.	False
origin_visible	bool	Ensure that the top left of the widget is within the window.	True
force	bool	Force scrolling even when prohibited by overflow styling.	False

Name	Туре	Description	Default
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
x_axis	bool	Allow scrolling on X axis?	True
y_axis	bool	Allow scrolling on Y axis?	True
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

Returns:

Туре	Description
Offset	The distance that was scrolled.

meth scroll_to_widget

```
scroll_to_widget(
    widget,
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    center=False,
    top=False,
    origin_visible=True,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False
```

Scroll scrolling to bring a widget into view.

Name	Туре	Description	Default
widget	Widget	A descendant widget.	required
animate	bool	True to animate, or False to jump.	True

Name	Туре	Description	Default
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
top	bool	Scroll widget to top of container.	False
origin_visible	bool	Ensure that the top left of the widget is within the window.	True
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

Returns:

Туре	Description
bool	True if any scrolling has occurred in any descendant, otherwise False.

meth scroll_up

```
scroll_up(
    *,
    animate=True,
    speed=None,
    duration=None,
    easing=None,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False
)
```

Scroll one line up.

Parameters:

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

meth scroll_visible

```
scroll_visible(
    animate=True,
    *,
    speed=None,
    duration=None,
    top=False,
    easing=None,
    force=False,
    on_complete=None,
    level="basic",
    immediate=False
)
```

Scroll the container to make this widget visible.

Name	Type	Description	Default

Name	Туре	Description	Default
animate	bool	Animate scroll.	True
speed	float None	Speed of scroll if animate is True; Or None to use duration.	None
duration	float None	Duration of animation, if animate is True and speed is None.	None
top	bool	Scroll to top of container.	False
easing	EasingFunction str None	An easing method for the scrolling animation.	None
force	bool	Force scrolling even when prohibited by overflow styling.	False
on_complete	CallbackType None	A callable to invoke when the animation is finished.	None
level	AnimationLevel	Minimum level required for the animation to take place (inclusive).	'basic'
immediate	bool	If False the scroll will be deferred until after a screen refresh, set to True to scroll immediately.	False

meth selection_updated

selection_updated(selection)

Called when the selection is updated.

Parameters:

Name	Туре	Description	Default
selection	Selection None	Selection information or None if no selection.	required

meth set_loading

set_loading(loading)

Set or reset the loading state of this widget.

A widget in a loading state will display a LoadingIndicator or a custom widget set through overriding the

get_loading_widget method.

Parameters:

Name	Туре	Description	Default
loading	bool	True to put the widget into a loading state, or False to reset the loading state.	required

meth set_scroll

set_scroll(x, y)

Set the scroll position without any validation.

This is a low-level method for when you want to see the scroll position in the next frame. For a more fully featured method, see scroll_to.

Parameters:

Name	Туре	Description	Default
х	float None	Desired x coordinate.	required
у	float None	Desired Y coordinate.	required

meth stop_animation

async

stop_animation(attribute, complete=True)

Stop an animation on an attribute.

Parameters:

Name	Туре	Description	Default
attribute	str	Name of the attribute whose animation should be stopped.	required
complete	bool	Should the animation be set to its final value?	True



💋 Note

If there is no animation scheduled or running, this is a no-op.

```
suppress_click()
   Suppress a click event.
   This will prevent a Click event being sent, if called after a mouse down event and before the click itself.
meth text_select_all
 text_select_all()
   Select the entire widget.
meth watch_disabled
 watch_disabled(disabled)
   Update the styles of the widget and its children when disabled is toggled.
meth watch_has_focus
 watch_has_focus(value)
   Update from CSS if has focus state changes.
meth watch_mouse_hover
 watch_mouse_hover(value)
   Update from CSS if mouse over state changes.
meth with_tooltip
 with_tooltip(tooltip)
   Chainable method to set a tooltip.
         Example
       def compose(self) -> ComposeResult:
           yield Label("Hello").with_tooltip("A greeting")
   Parameters:
      Name
                                                         Description
                                                                                                Default
                     Type
```

Name	Туре	Description	Default
tooltip	Visual RenderableType None	New tooltip, or None to clear the tooltip.	required

Returns:

Туре	Description
Self	Self.

class WidgetError

Bases: Exception

Base widget error.

() July 17, 2024