Gnome::Gtk3::Paned

Paned — A widget with two adjustable panes

Table of Contents

```
unit class Gnome::Gtk3::Paned;
also is Gnome::Gtk3::Container;
```

Synopsis

```
my Gnome::Gtk3::Paned $p .= new(:orientation(GTK_ORIENTATION_HORIZONTAL));
my Gnome::Gtk3::ListBox $lb1 .= new(:empty);
my Gnome::Gtk3::ListBox $lb2 .= new(:empty);
$p.gtk-paned-add1($lb1);
$p.gtk-paned-add2($lb2);
```

Methods

new

```
multi method new ( :$orientation! )
```

Create a new object with an orientation set to `GTK_ORIENTATION_HORIZONTAL` or `GTK_ORIENTATION_VERTICAL`.

```
multi method new ( :$widget! )
```

Create an object using a native object from elsewhere. See also Gnome::GObject::Object.

```
multi method new ( Str :$build-id! )
```

Create an object using a native object from a builder. See also Gnome::GObject::Object.

gtk_paned_new

Creates a new native pane

```
method gtk_paned_new ( Int $orientation --> N-GObject )
```

Returns a native widget. Can be used to initialize another object using :widget. This is very cumbersome when you know that a oneliner does the job for you: `my Gnome::Gtk3::Paned $m .= new(:$orientation);

```
my Gnome::Gtk3::Paned $m;
$m .= :new(:widget($m.gtk_paned_new(GTK_ORIENTATION_HORIZONTAL));
```

gtk_paned_add1

Adds a child to the top or left pane with default parameters. This is equivalent to `$p.gtk_paned_pack1( $child-widget, 0, 1)`.

```
method gtk_paned_add1 ( N-GObject $child-widget )
```

- $child-widget; Native child widget to add. When a Perl6 widget object is provided, the native widget is retrieved from that object. See example in the synopsis.

gtk_paned_add2

Adds a child to the bottom or right pane with default parameters. This is equivalent to `$p.gtk_paned_pack2( $child-widget, 1, 1)`.

```
method gtk_paned_add2 ( N-GObject $child-widget )
```

- $child-widget; Native child widget to add. When a Perl6 widget object is provided, the native widget is retrieved from that object. See example in the synopsis.

gtk_paned_pack1

Adds a child to the top or left pane.

```
method gtk_paned_pack1 ( N-GObject $child, Int $resize, Int $shrink )
```

- $child; the child to add. When a Perl6 widget object is provided, the native widget is retrieved from that object. See example in the synopsis.
- $resize; should this child expand when the paned widget is resized.
- $shrink; can this child be made smaller than its requisition.

gtk_paned_pack2

Adds a child to the bottom or right pane.

```
method gtk_paned_pack2 ( N-GObject $child, Int $resize, Int $shrink )
```

- $child; the child to add. When a Perl6 widget object is provided, the native widget is retrieved from that object. See example in the synopsis.
- $resize; should this child expand when the paned widget is resized.
- $shrink; can this child be made smaller than its requisition.

[gtk_paned_] get_child1

Obtains the first child of the paned widget.

```
method gtk_paned_get_child1 ( --> N-GObject )
```

Returns the first child

```
my Gnome::Gtk3::ListBox $lb .= new(:widget($p.get-child1()));
```

[gtk_paned_] get_child2

Obtains the second child of the paned widget.

```
method gtk_paned_get_child2 ( --> N-GObject )
```

Returns the second child

[gtk_paned_] set_position

Sets the position of the divider between the two panes.

```
method gtk_paned_set_position ( Int $position )
```

- $position; pixel position of divider, a negative value means that the position is unset.

[gtk_paned_] get_position

Obtains the position of the divider between the two panes.

```
method gtk_paned_get_position ( --> Int )
```

Returns position of the devider.

[gtk_paned_] get_handle_window

Returns the `GdkWindow` of the handle. This function is useful when handling button or motion events because it enables the callback to distinguish between the window of the paned, a child and the handle.

```
method gtk_paned_get_handle_window ( --> N-GObject )
```

Returns a native `GdkWindow`

[gtk_paned_] set_wide_handle

Sets the “wide-handle” property.

```
method gtk_paned_set_wide_handle ( Int $wide )
```

- $wide; the new value for the “wide-handle” property, this is 0 or 1.

[gtk_paned_] get_wide_handle

Gets the “wide-handle” property.

```
method gtk_paned_get_wide_handle ( --> Int )
```

Returns 0 or 1 for the wide-handle property.

Signals

Supported signals

accept-position

The `accept-position` signal is a keybinding signal which gets emitted to accept the current position of the handle when moving it using key bindings.

The default binding for this signal is **Return** or **Space**.

cancel-position

The `cancel-position` signal is a keybinding signal which gets emitted to cancel moving the position of the handle using key bindings. The position of the handle will be reset to the value prior to moving it.

The default binding for this signal is **Escape**.

toggle-handle-focus

The `toggle-handle-focus` is a keybinding signal which gets emitted to accept the current position of the handle and then move focus to the next widget in the focus chain.

The default binding is **Tab**.

Not yet supported signals

cycle-child-focus

The `cycle-child-focus` signal is a keybinding signal which gets emitted to cycle the focus between the children of the paned.

The default binding is **f6**.

cycle-handle-focus

The `cycle-handle-focus` signal is a keybinding signal which gets emitted to cycle whether the paned should grab focus to allow the user to change position of the handle by using key bindings.

The default binding for this signal is **f8**.

move-handle

The `move-handle` signal is a keybinding signal which gets emitted to move the handle when the user is using key bindings to move it.