NSSliderCell

Inherits From:	NSActionCell : NSCell : NSObject
Conforms To:	NSCoding, NSCopying (from NSCell) NSObject (from NSObject)
Declared In:	AppKit/NSSliderCell.h

Class Description

An NSSliderCell controls the appearance and behavior of an NSSlider, or of a single slider in an NSMatrix.

You can customize an NSSliderCell to a certain degree, using its **set...** methods. If these do not allow you sufficient flexibility, you can create a subclass. In that subclass, you can override any of the following methods: **knobRectFlipped:**, **drawBarInside:flipped:**, **drawKnob:**, or **prefersTrackingUntilMouseUp**.

Method Types

Asking about the cell's behavior	– altIncrementValue + prefersTrackingUntilMouseUp – trackRect
Changing the cell's behavior	– setAltIncrementValue:
Displaying the cell	– knobRectFlipped: – drawBarInside:flipped: – drawKnob – drawKnob:
Asking about the cell's appearance	– knobThickness – isVertical – title – titleCell – titleFont – titleColor

Changing the cell's appearance	– setKnobThickness: – setTitle: – setTitleCell: – setTitleColor: – setTitleFont:
Asking about the value limits	– maxValue – minValue
Changing the value limits	– setMaxValue – setMinValue

Class Methods

prefersTrackingUntilMouseUp

+ (BOOL)**prefersTrackingUntilMouseUp**

By default, this method returns YES, so that an NSSliderCell continues to track the mouse even after the mouse leaves the cell's tracking rectangle. This means that, once you take hold of a slider's knob (by putting the mouse inside the cell's frame rectangle and pressing the mouse button), you retain control of the knob until you release the mouse button, even if you drag the mouse clear to the other side of the screen.

Never call this method explicitly. Override it if you create a subclass of NSSliderCell that you want to track the mouse differently.

Instance Methods

altIncrementValue

– (double)**altIncrementValue**

Returns the amount that the slider will change its value when the user drags the knob with the Alt key held down.

Unless you call **setAltIncrementValue**, **altIncrementValue** returns –1.0, and the slider behaves no differently with the Alt key down than with it up.

See also: **– setAltIncrementValue:**

drawBarInside:flipped:

– (void)**drawBarInside:**(NSRect)*aRect* **flipped:**(BOOL)*flipped*

Draws the slider's bar—but not its bezel or knob—in *aRect*.

flipped indicates whether the cell's control view—that is, the NSSlider or NSMatrix associated with the NSSliderCell—has a flipped coordinate system.

You should never invoke this method explicitly. It's included so that you can override it in a subclass.

See also: – **drawKnob:**

drawKnob

– (void)**drawKnob**

Calculates the rectangle in which the knob should be drawn, then invokes **drawKnob:** to actually draw the knob. Before this message is sent, a **lockFocus** method must be sent to the cell's control view.

You might invoke this method if you override one of the display methods belonging to NSControl or NSCell.

If you create a subclass of NSSliderCell, don't override this method. Override **drawKnob:** instead.

drawKnob:

– (void)**drawKnob:**(NSRect)*knobRect*

Draws the knob in *knobRect*. Before this message is sent, a **lockFocus** message must be sent to the cell's control view.

You should never invoke this method explicitly. It's included so that you can override it in a subclass.

isVertical

– (int)**isVertical**

Returns 1 if the slider is vertical, 0 if it's horizontal, and –1 if the orientation can't be determined (for example, if the slider hasn't been displayed yet). A slider is defined as vertical if its height is greater than its width.

knobRectFlipped:

– (NSRect)**knobRectFlipped:**(BOOL)*flipped*

Returns the rectangle in which the knob will be drawn, specified in the coordinate system of the NSSlider or NSMatrix with which the NSSliderCell is associated. *flipped* indicates whether that coordinate system is flipped, a question you can answer by sending NSView's **isFlipped** message to the NSMatrix or NSSlider.

The knob rectangle depends on where in the slider the knob belongs—that is, it depends on the SliderCell's minimum and maximum values, and on the value which the position of the knob will represent.

You should never invoke this method explicitly. It's included so that you can override it in a subclass.

knobThickness

– (float)**knobThickness**

Returns the knob's thickness, in pixels. The thickness is defined to be the extent of the knob along the long dimension of the bar. In a vertical slider, then, a knob's thickness is its height; in a horizontal slider, its thickness is its width.

See also: – **setKnobThickness:**

maxValue

– (double)**maxValue**

Returns the maximum value that the slider can send to its target. A horizontal slider sends its maximum value when the knob is at the right end of the slider; a vertical slider sends it when the knob is at the top.

See also: – **setMaxValue:**

minValue

– (double)**minValue**

Returns the minimum value that the slider can send to its target. A vertical slider sends this value when its knob is at the bottom; a horizontal slider sends it when its knob is all the way to the left.

setAltIncrementValue:

– (void)**setAltIncrementValue:**(double)*increment*

Sets the amount by which the NSSliderCell modifies its value when the knob is Alt-dragged. *increment* should fit the range of values that the slider can represent—for example, if the slider has a minimum value of 5 and a maximum value of 10, *increment* should be between 0 and 5.

If you don't call this method, the slider behaves the same with the Alt key down as with it up. This is also the result when you call **setAltIncrementValue:** with an *increment* of -1.

See also: – **maxValue**, – **minValue**

setKnobThickness:

– (void)**setKnobThickness:**(float)*thickness*

Lets you set the knob's thickness, measured in pixels. The thickness is defined to be the extent of the knob along the long dimension of the bar. In a vertical slider, then, a knob's thickness is its height; in a horizontal slider, its thickness is its width.

See also: – **knobThickness**

setMaxValue:

– (void)**setMaxValue:**(double)*aDouble*

Sets the maximum value that the slider can send to its target—the value that a horizontal slider will send when its knob is all the way to the right, or that a vertical slider will send when its knob is at the top.

See also: – **maxValue**

setMinValue:

– (void)**setMinValue:**(double)*aDouble*

Sets the minimum value that the slider can send to its target. A horizontal slider sends its minimum value when its knob is all the way to the left; a vertical slider sends its minimum value when its knob is at the bottom.

See also: – **minValue**

setTitle:

– (void)**setTitle:**(NSString *)*title*

Sets the title in the bar behind the slider's knob to *title*.

See also: – **title**

setTitleCell:

– (void)**setTitleCell:**(NSCell *)*aCell*

Sets the cell used to draw the slider's title. *You only need to invoke this method if the default title cell, NSTextFieldCell, doesn't suit your needs—that is, if you want to display the title in a manner that*

NSTextFieldCell doesn't permit. When you do choose to override the default, aCell should be an instance of a subclass of TextFieldCell.

See also: – **titleCell**

setTitleColor:

– (void)**setTitleColor:**(NSColor *)*color*

Sets the color used to draw the slider's title.

See also: – **titleColor**

setTitleFont:

– (void)**setTitleFont:**(NSFont *)*font*

Sets the font used to draw the slider's title.

See also: – **titleFont**

title

– (NSString *)**title**

Returns the slider's title. The default title is the empty string ("").

See also: – **setTitle:**

titleCell

– (id)**titleCell**

Returns the cell used to draw the title. The default is an NSTextFieldCell.

See also: – **setTitleCell:**

titleColor

– (NSColor *)**titleColor**

Returns the color used to draw the slider's title. The default color is NSColor's **controlTextColor**.

See also: – **setTitleColor:**

titleFont

– (NSFont *)**titleFont**

Returns the font used to draw the slider's title.

See also: – **setTitleFont:**

trackRect

– (NSRect)**trackRect**

Returns the rectangle within which the cell tracks the mouse while the mouse button is down. This rectangle includes the slider bar, but not the bezel.