QSlider Class

QSlider 小部件提供垂直或水平滑块。更多的...

Header:	**#include**
CMake:	find_package(Qt6 REQUIRED COMPONENTS Widgets) target_link_libraries(mytarget PRIVATE Qt6::Widgets)
qmake:	QT += widgets
Inherits:	QAbstractSlider

- 所有成员的列表，包括继承的成员

公共类型

enum **TickPosition { NoTicks, TicksBothSides, TicksAbove, TicksBelow, TicksLeft, TicksRight }**

特性

- **tickInterval** : int
- **tickPosition** : TickPosition

公共职能

	QSlider(QWidget *parent = nullptr)
	QSlider(Qt::Orientation *orientation*, QWidget *parent = nullptr)
virtual	~QSlider()
void	setTickInterval(int *ti*)
void	setTickPosition(QSlider::TickPosition *position*)
int	tickInterval() const
QSlider::TickPosition	tickPosition() const

重载的公共职能

virtual bool	**event(QEvent **event*) override**
virtual QSize	**minimumSizeHint**() const override
virtual QSize	**sizeHint**() const override

protected功能

virtual void	**initStyleOption(QStyleOptionSlider **option*) const**

重载的protected功能

virtual void	**mouseMoveEvent(QMouseEvent **ev*) override**
virtual void	**mousePressEvent**(QMouseEvent **ev*) override
virtual void	**mouseReleaseEvent**(QMouseEvent **ev*) override
virtual void	**paintEvent**(QPaintEvent **ev*) override

详细说明

滑块是用于控制有界值的经典小部件。它允许用户沿着水平或垂直凹槽移动滑块手柄，并将手柄的位置转换为合法范围内的整数值。

QSlider本身的功能很少；大部分功能都在QAbstractSlider。最有用的功能是setValue() 将滑块直接设置为某个值；triggerAction() 模拟点击的效果（对快捷键有用）；setSingleStep(),setPageStep() 设置步骤；和setMinimum（） 和setMaximum() 定义滚动条的范围。

QSlider 提供了控制刻度线的方法。您可以使用setTickPosition() 指示您想要刻度线的位置，setTickInterval() 表示您想要多少个。可以使用以下命令查询当前设置的刻度位置和间隔tickPosition（） 和tickInterval() 函数分别。

QSlider继承了一套全面的信号：

信号	描述
valueChanged()	当滑块的值发生更改时发出。Tracking() 确定在用户交互期间是否发出该信号。
sliderPressed()	当用户开始拖动滑块时发出。
sliderMoved()	当用户拖动滑块时发出。
sliderReleased()	当用户释放滑块时发出。

QSlider 仅提供整数范围。请注意，虽然 QSlider 可以处理非常大的数字，但用户很难在非常大的范围内准确地使用滑块。

滑块接受 Tab 上的焦点并提供鼠标滚轮和键盘界面。键盘接口如下：

- 向左/向右将水平滑块移动一步。
- 向上/向下将垂直滑块移动一步。
- PageUp 向上移动一页。
- PageDown 向下移动一页。
- Home 移动到开头（最小）。
- End 移动到末尾（最大）。

也可以看看QScrollBar,QSpinBox,QDial， 和Sliders Example。

会员类型文档

enum QSlider::TickPosition

该枚举指定相对于滑块凹槽和用户移动手柄的刻度线的绘制位置。

持续的	价值	描述
`QSlider::NoTicks`	`0`	不要画任何刻度线。
`QSlider::TicksBothSides`	`3`	在凹槽两侧画刻度线。
`QSlider::TicksAbove`	`1`	在（水平）滑块上方绘制刻度线
`QSlider::TicksBelow`	`2`	在（水平）滑块下方绘制刻度线
`QSlider::TicksLeft`	`TicksAbove`	在（垂直）滑块左侧绘制刻度线
`QSlider::TicksRight`	`TicksBelow`	在（垂直）滑块右侧绘制刻度线

属性文档

tickInterval : int

该属性保存刻度线之间的间隔

这是一个值间隔，而不是像素间隔。如果为 0，滑块将在 singleStep 和 pageStep 之间进行选择。

默认值为 0。

访问功能：

int	**tickInterval() const**
void	**setTickInterval**(int *ti*)

也可以看看tickPosition,singleStep， 和pageStep。

tickPosition : TickPosition

该属性保存该滑块的刻度线位置

有效值由以下描述QSlider::TickPosition枚举。

默认值为QSlider::NoTicks。

访问功能：

QSlider::TickPosition	tickPosition() const
void	**setTickPosition**(QSlider::TickPosition *position*)

也可以看看tickInterval。

成员函数文档

*[explicit]QSlider::QSlider(QWidget *parent = nullptr)*

使用给定的构造一个垂直滑块*parent*。

*[explicit]QSlider::QSlider(Qt::Orientation orientation, QWidget *parent = nullptr)*

使用给定的构造一个滑块*parent*。这*orientation*参数决定滑块是水平还是垂直；有效值为Qt::Vertical和Qt::Horizontal。

[virtual]QSlider::~QSlider()

销毁该滑块。

*[override virtual]bool QSlider::event(QEvent *event)*

重新实现：QAbstractSlider::event（QEvent *e）。

*[virtual protected]void QSlider::initStyleOption(QStyleOptionSlider *option) const*

初始化*option*与此值QSlider。当子类需要时，此方法非常有用QStyleOptionSlider，但不想自己填写所有信息。

也可以看看QStyleOption::initFrom()。

[override virtual]QSize QSlider::minimumSizeHint() const

重新实现属性的访问函数：QWidget::minimumSizeHint。

*[override virtual protected]void QSlider::mouseMoveEvent(QMouseEvent *ev)*

重新实现：QWidget::mouseMoveEvent（QMouseEvent *事件）。

*[override virtual protected]void QSlider::mousePressEvent(QMouseEvent *ev)*

重新实现：QWidget::mousePressEvent（QMouseEvent *事件）。

*[override virtual protected]void QSlider::mouseReleaseEvent(QMouseEvent *ev)*

重新实现：QWidget::mouseReleaseEvent（QMouseEvent *事件）。

*[override virtual protected]void QSlider::paintEvent(QPaintEvent *ev)*

重新实现：QWidget::paintEvent（QPaintEvent *事件）。

[override virtual]QSize QSlider::sizeHint() const

重新实现属性的访问函数：QWidget::sizeHint。