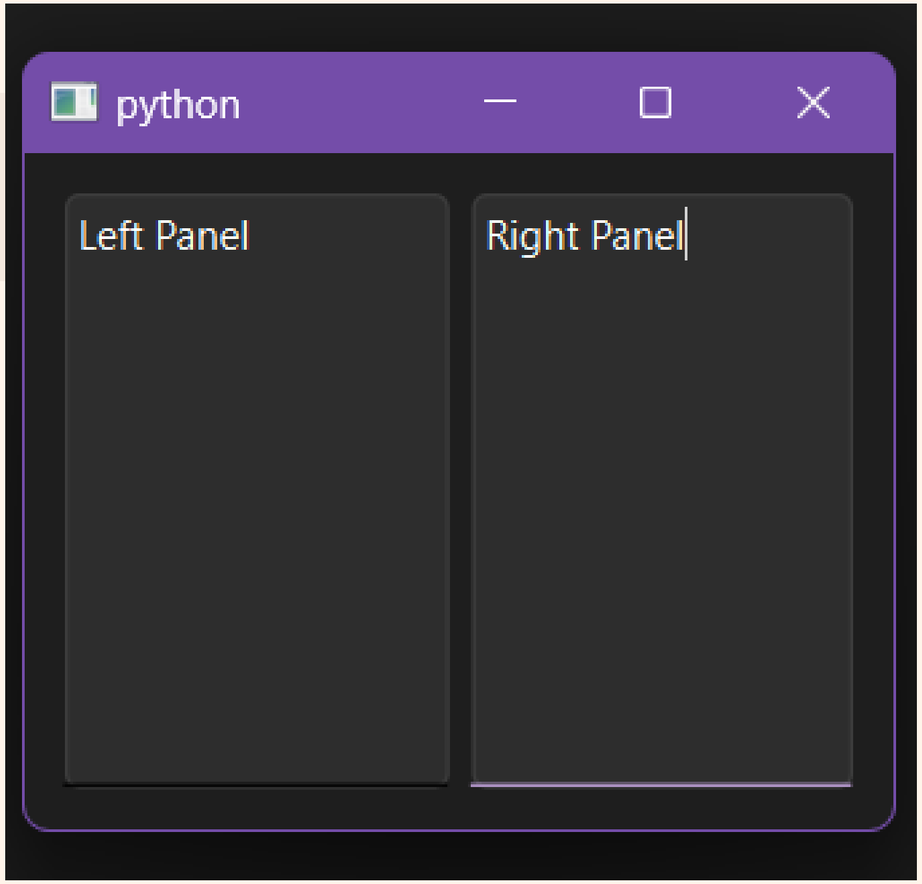
WEEK 1:

ADVANCED GUI ATTRIBUTES & METHODS

AMADOR, ORLINO, PALACIOS, TAPA

QSplitter



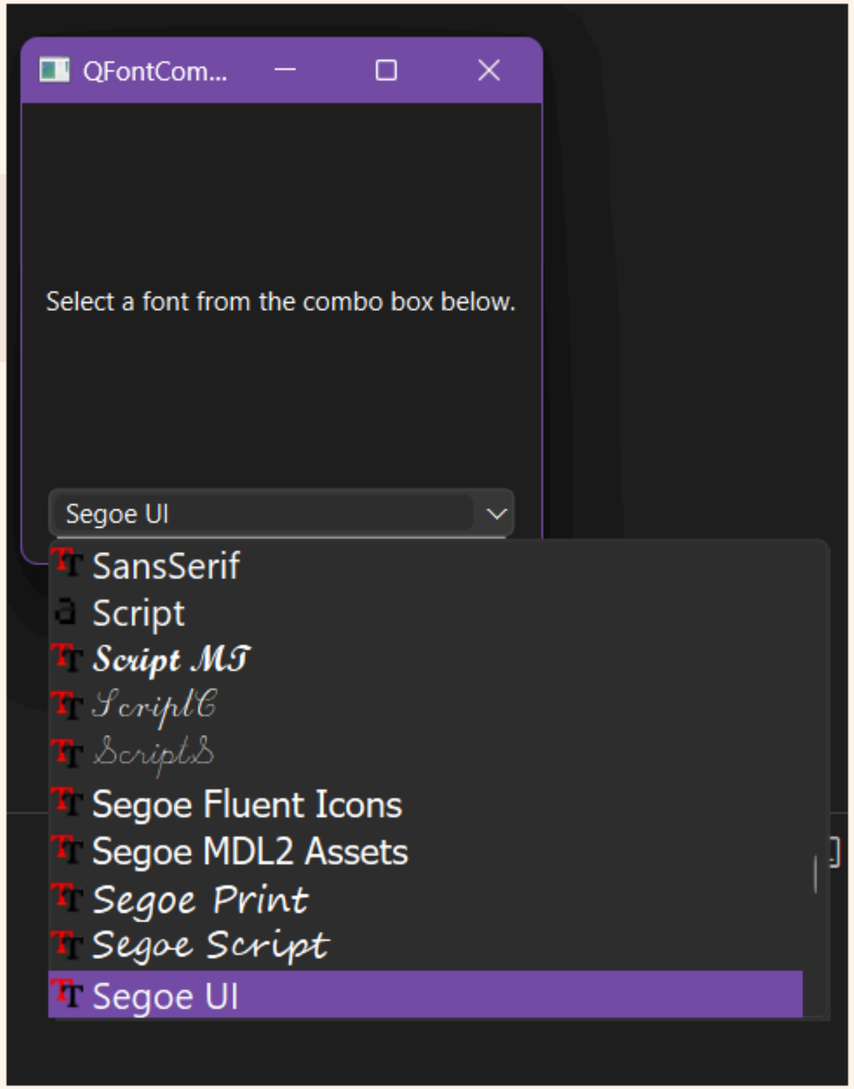
Definition

is a widget in Qt that allows you to divide a space into resizable sections.

Code

```python
from PySide6.QtWidgets import QApplication, QWidget, QSplitter, QTextEdit, QVBoxLayout
import sys

app = QApplication(sys.argv)

# Create main window
window = QWidget()
layout = QVBoxLayout(window)

# Create a splitter
splitter = QSplitter()

# Add widgets to the splitter
text1 = QTextEdit("Left Panel")
text2 = QTextEdit("Right Panel")
splitter.addWidget(text1)
splitter.addWidget(text2)

layout.addWidget(splitter)
window.show()

sys.exit(app.exec())
```

QFontComboBox



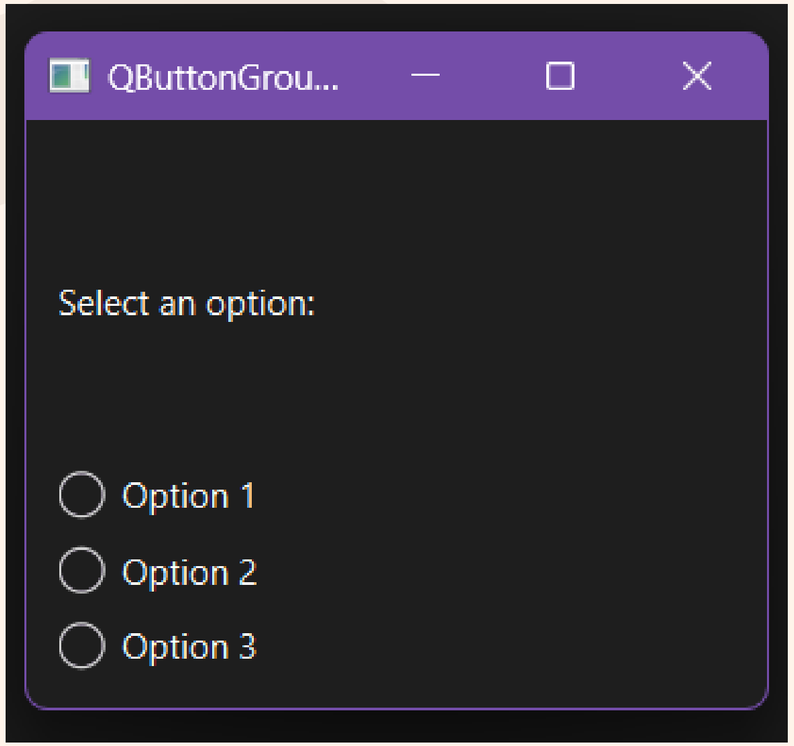
Definition

is a widget in Qt that provides a dropdown list of available fonts, allowing users to select a font easily.

Code

```
class FontComboBoxExample(QWidget):
    def __init__(self):
        super().__init__()

        self.setWindowTitle("QFontComboBox Example")
        self.setGeometry(100, 100, 400, 300)

        # Create a layout
        layout = QVBoxLayout()

        # Create label to display the selected font
        self.label = QLabel("Select a font from the combo box below.")

        # Create QFontComboBox to display font options
        self.font_combo = QFontComboBox()

        # Connect font selection signal to a custom slot
        self.font_combo.currentFontChanged.connect(self.on_font_changed)

        # Add the label and combo box to the layout
```

QButtonGroup



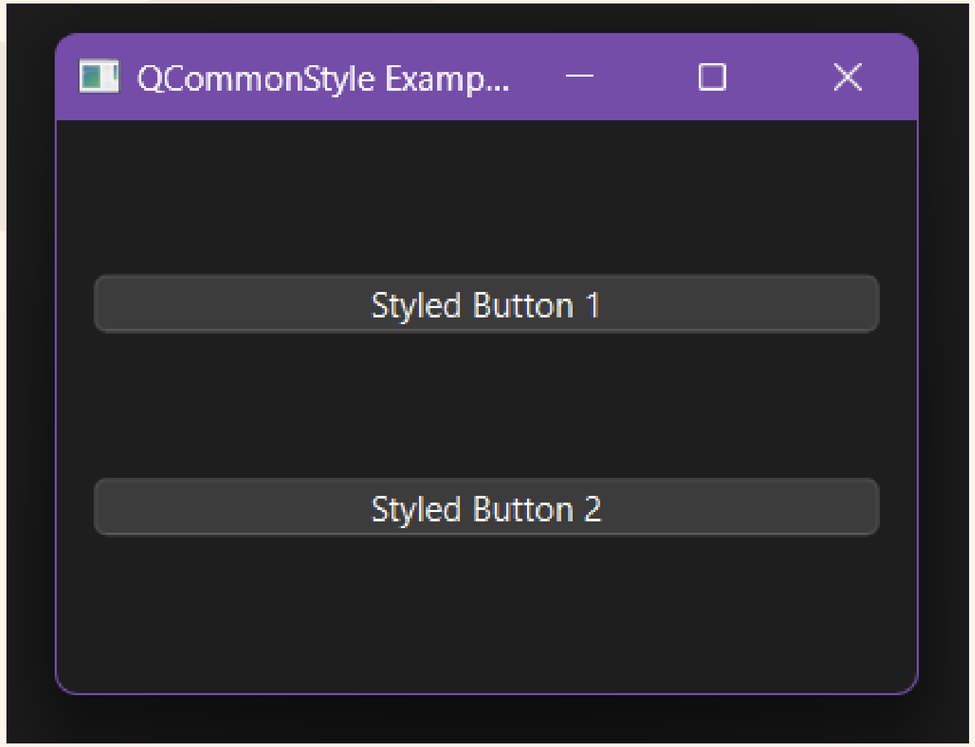
Definition

is a Qt class that groups multiple buttons together, ensuring only one button in the group can be selected at a time, typically used with radio buttons or checkboxes.

Code

```
# Create label to show selected option
self.label = QLabel("Select an option:")

# Create a button group
self.button_group = QButtonGroup(self)

# Create radio buttons
self.radio1 = QRadioButton("Option 1")
self.radio2 = QRadioButton("Option 2")
self.radio3 = QRadioButton("Option 3")

# Add buttons to the group
self.button_group.addButton(self.radio1, 1)
self.button_group.addButton(self.radio2, 2)
self.button_group.addButton(self.radio3, 3)

# Connect signal when a button is clicked
self.button_group.buttonClicked.connect(self.on_button_clicked)

# Add widgets to layout
```

QCommonStyle



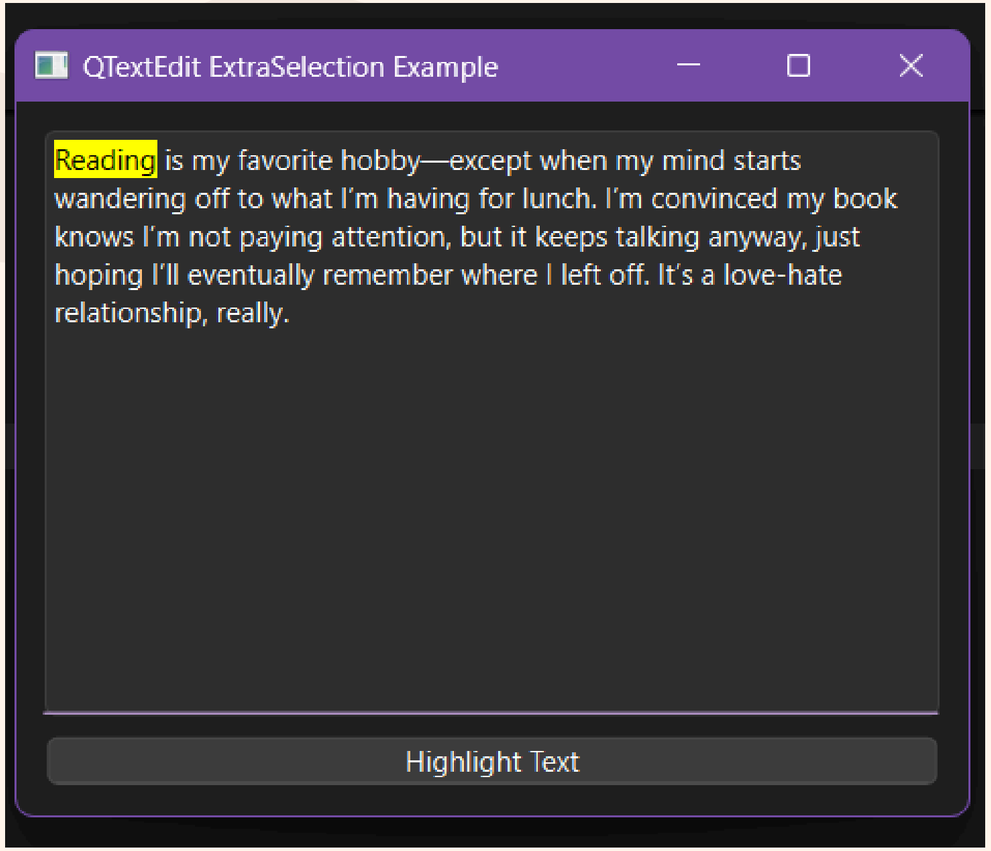
Definition

provides a standard, native look for widgets across platforms.

Code

```
        super().__init__()

        self.setWindowTitle("QCommonStyle Example")
        self.setGeometry(100, 100, 300, 200)

        # Create a layout
        layout = QVBoxLayout()

        # Create buttons
        button1 = QPushButton("Styled Button 1")
        button2 = QPushButton("Styled Button 2")

        # Create QCommonStyle instance
        common_style = QCommonStyle()

        # Apply styles to buttons
        common_style.polish(button1)
        common_style.polish(button2)

        # Add buttons to layout
        layout.addWidget(button1)
        layout.addWidget(button2)

        # Set layout
```

QExtraSelection



Definition

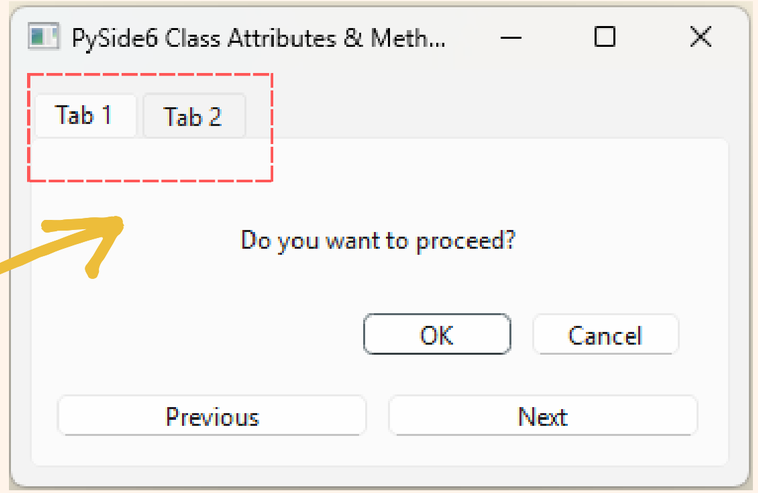
is a Qt class that groups multiple buttons together, ensuring only one button in the group can be selected at a time, typically used with radio buttons or checkboxes.

Code

```
def __init__(self):
    super().__init__()

    self.setWindowTitle("QTextEdit ExtraSelection Example")
    self.setGeometry(100, 100, 400, 300)

    # Create layout
    layout = QVBoxLayout()

    # Create QTextEdit widget
    self.text_edit = QTextEdit()

    # Create a button to highlight text
    highlight_button = QPushButton("Highlight Text")

    # Connect button click event to highlight function
    highlight_button.clicked.connect(self.highlight_text)

    # Add widgets to layout
    layout.addWidget(self.text_edit)
    layout.addWidget(highlight_button)

    # Set the layout for the window
    self.setLayout(layout)
```

```
def highlight_text(self):
    # Get the text cursor from QTextEdit
    cursor = self.text_edit.textCursor()

    # Check if the cursor has selected text
    if cursor.hasSelection():
        # Define a QTextCharFormat to specify how the
        extra_selection = QTextEdit.ExtraSelection()

        # Set text background color and text color
        text_format = QTextCharFormat()
        text_format.setBackground(QColor(255, 255, 0)
        text_format.setForeground(QColor(0, 0, 0))  #

        extra_selection.format = text_format
        extra_selection.cursor = cursor

        # Apply the extra selection to QTextEdit
        self.text_edit.setExtraSelections([extra_sele
```

QTabWidget

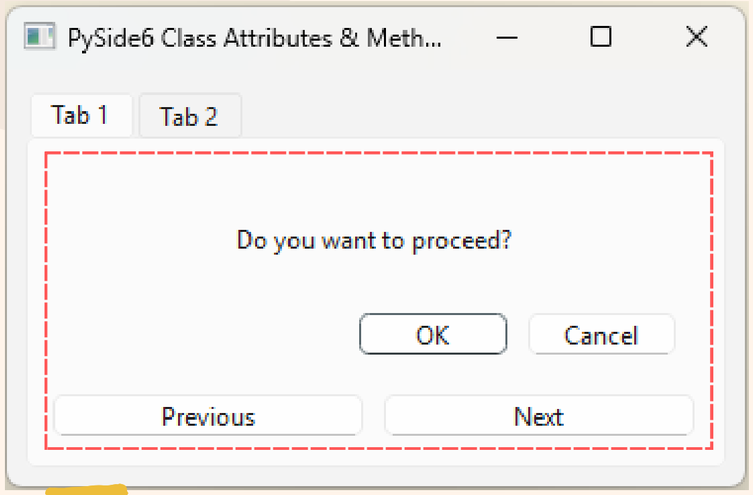


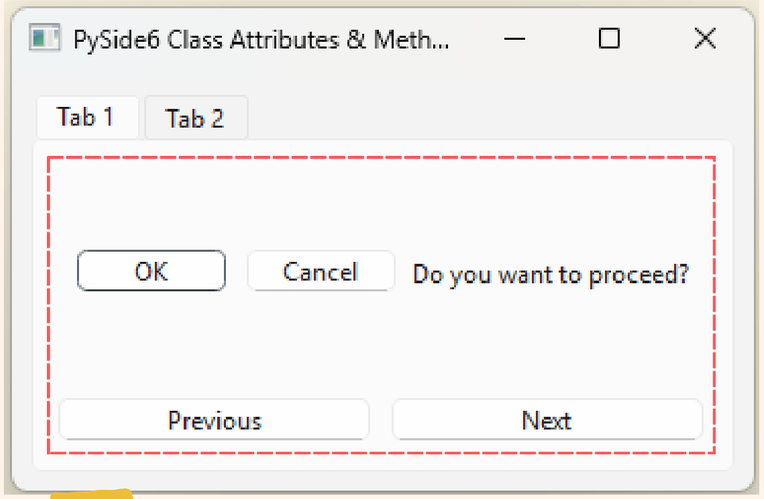
Attributes

- count() → Returns the number of tabs.
- currentIndex() → Gets the index of the currently selected tab.
- tabText(index) → Returns the tab's label text.

Methods

- addTab(widget, title) → Adds a new tab.
- setCurrentIndex(index) → Switches to a specific tab.
- removeTab(index) → Removes a tab.





QStackedWidget

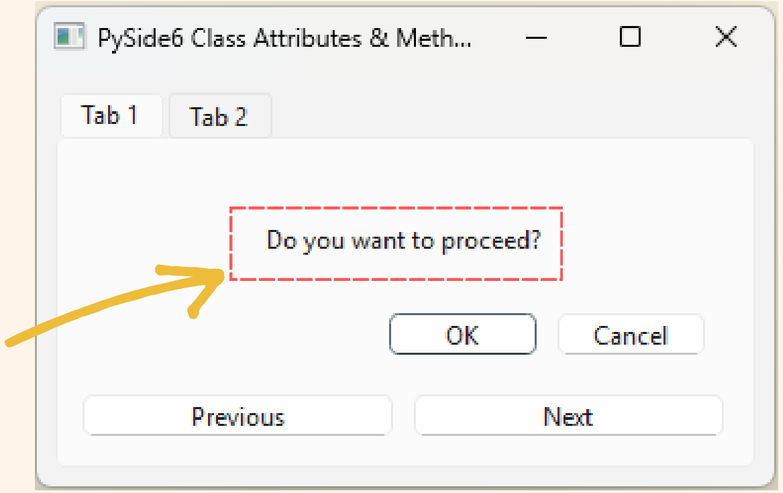
Attributes

- count() → Returns the number of pages.
- currentIndex() → Gets the index of the current page.

Methods

- addWidget(widget) → Adds a page to the stack.
- setCurrentIndex(index) → Switches to a specific page.

QLabel



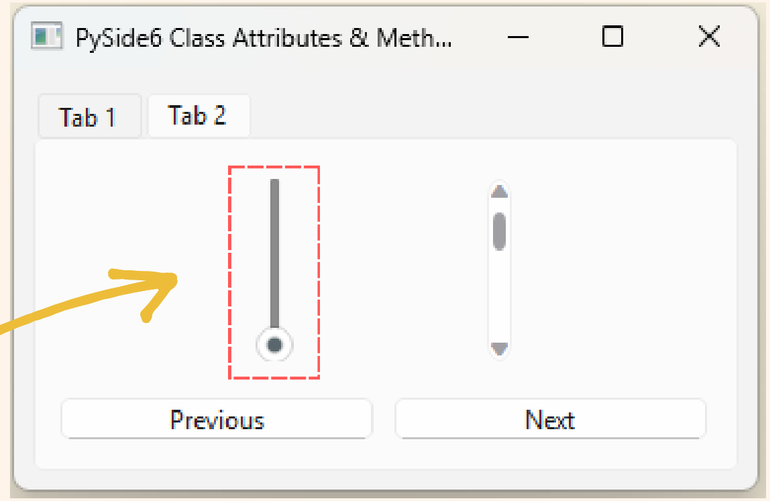
Attributes

- text() → Gets the label text.

Methods

- setText(text) → Sets the text.
- setAlignment(alignment) → Sets the alignment.

QSlider



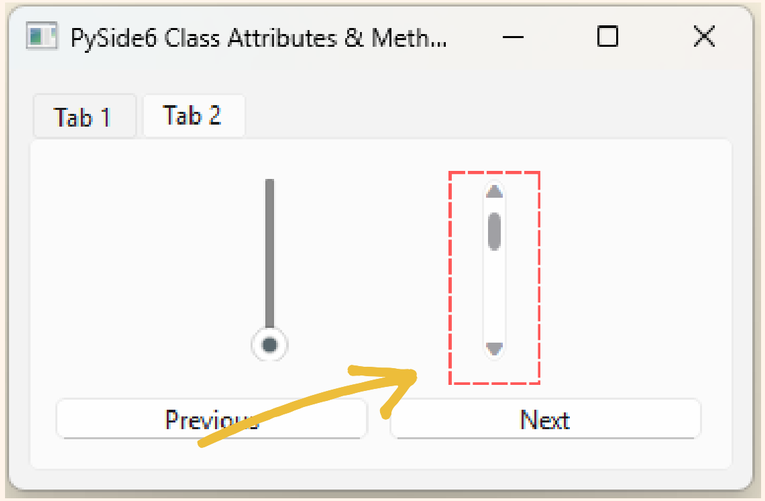
Attributes

- value() → Gets the slider's value.

Methods

- setValue(value) → Sets the slider's value.
- setOrientation(orientation) → Sets the slider's orientation (horizontal/vertical).

QScrollBar

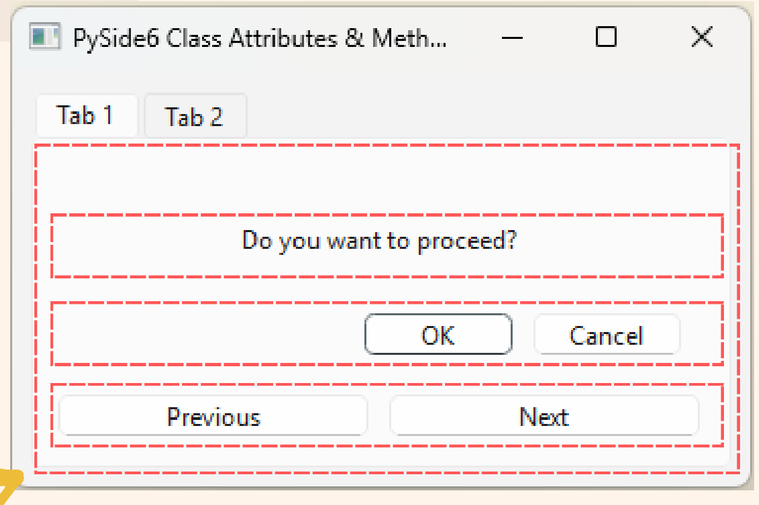


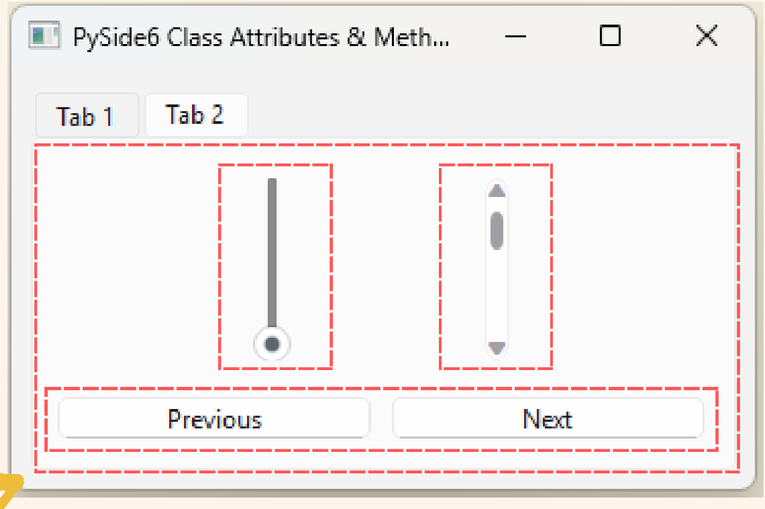
Attributes

- value() → Gets the scrollbar's value.

Methods

- setText(text) → Sets the text.
- setAlignment(alignment) → Sets the alignment.

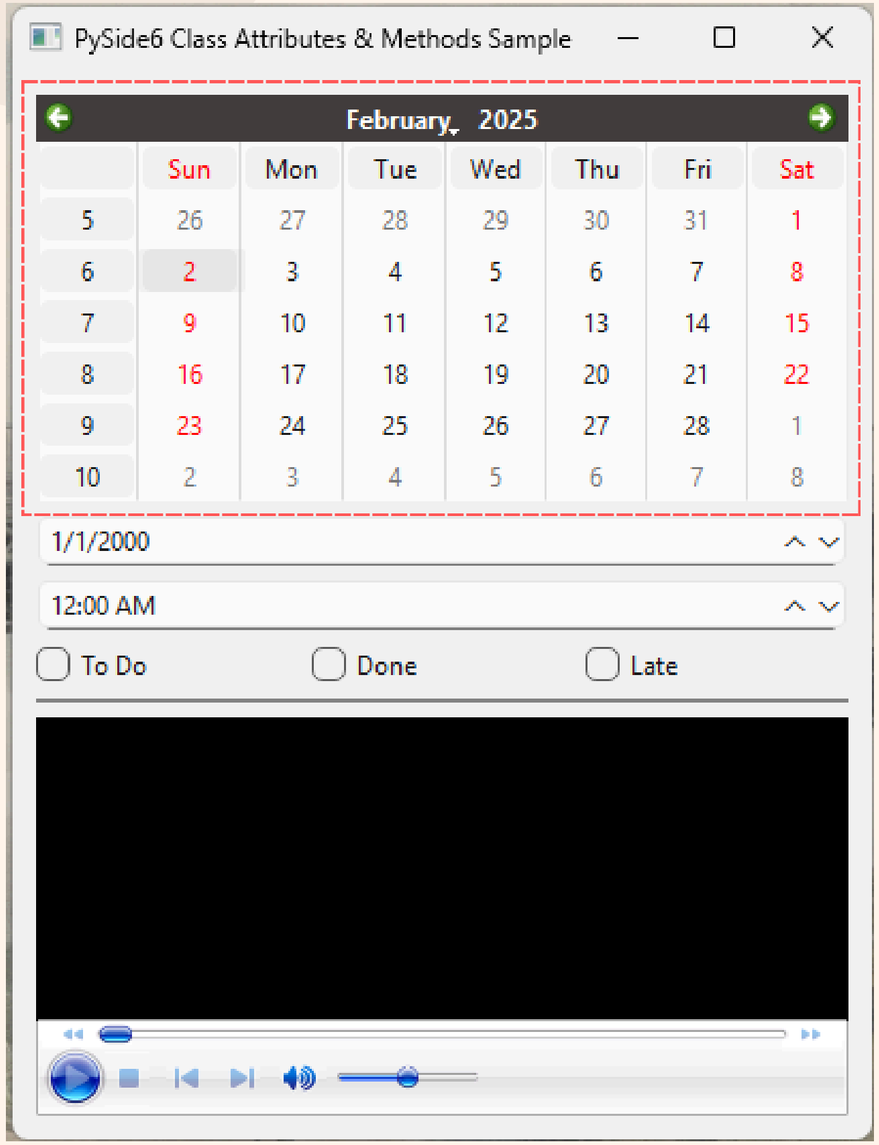




QGridLayout

Methods

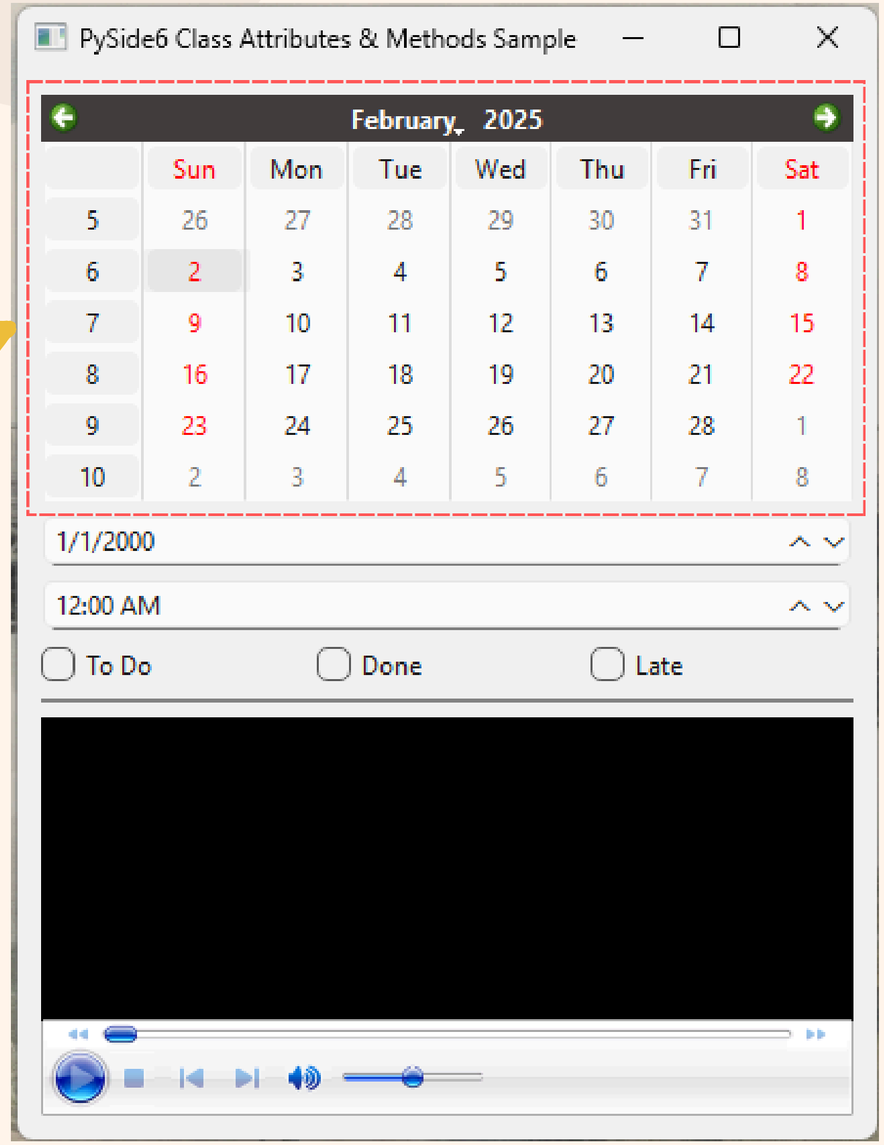
- addWidget(widget, row, col) → Adds a widget at a specific position.
- setSpacing(int) → Sets spacing between widgets.



QCalendarWidget

Methods

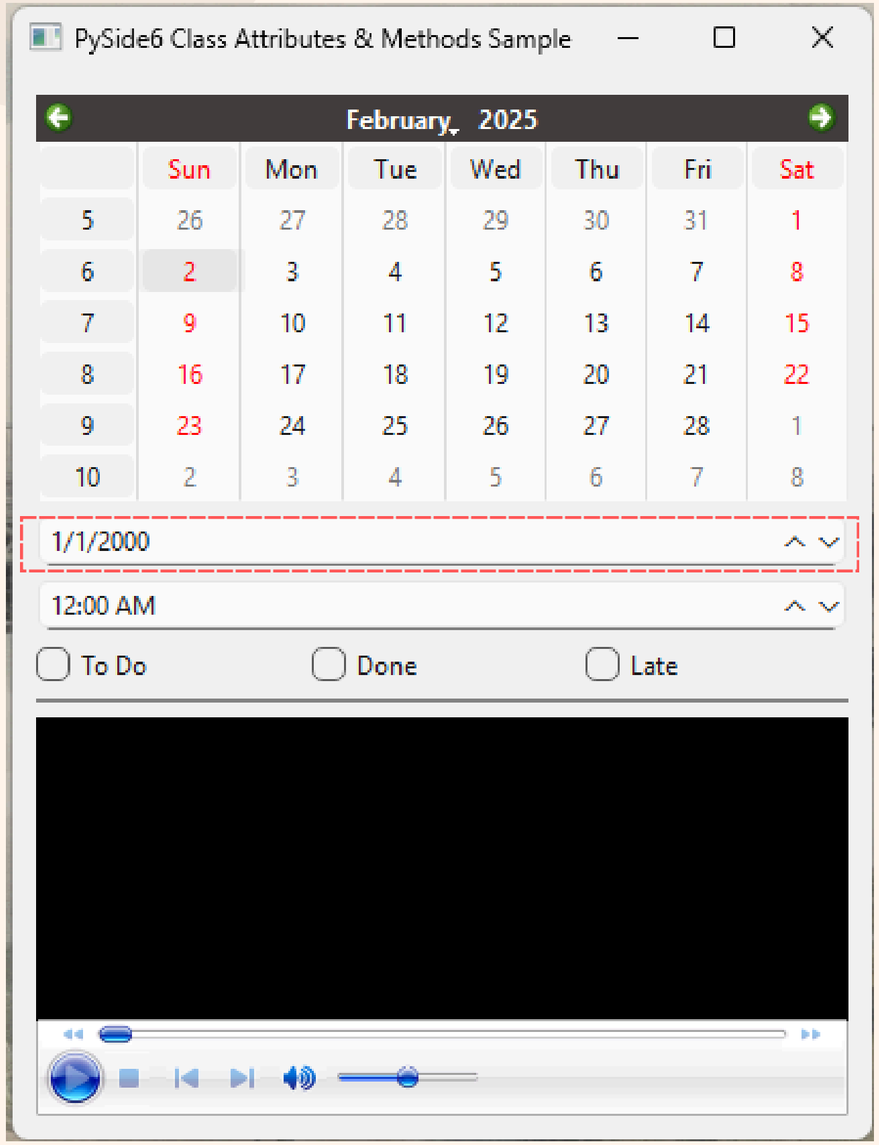
- selectedDate(): Returns the currently selected date.
- setSelectedDate(QDate): Sets the selected date.
- setGridVisible(bool): Shows or hides the grid.
- def __init__()
- def calendar()
- def clearMaximumDate()
- and more



QCalendarWidget

Attributes

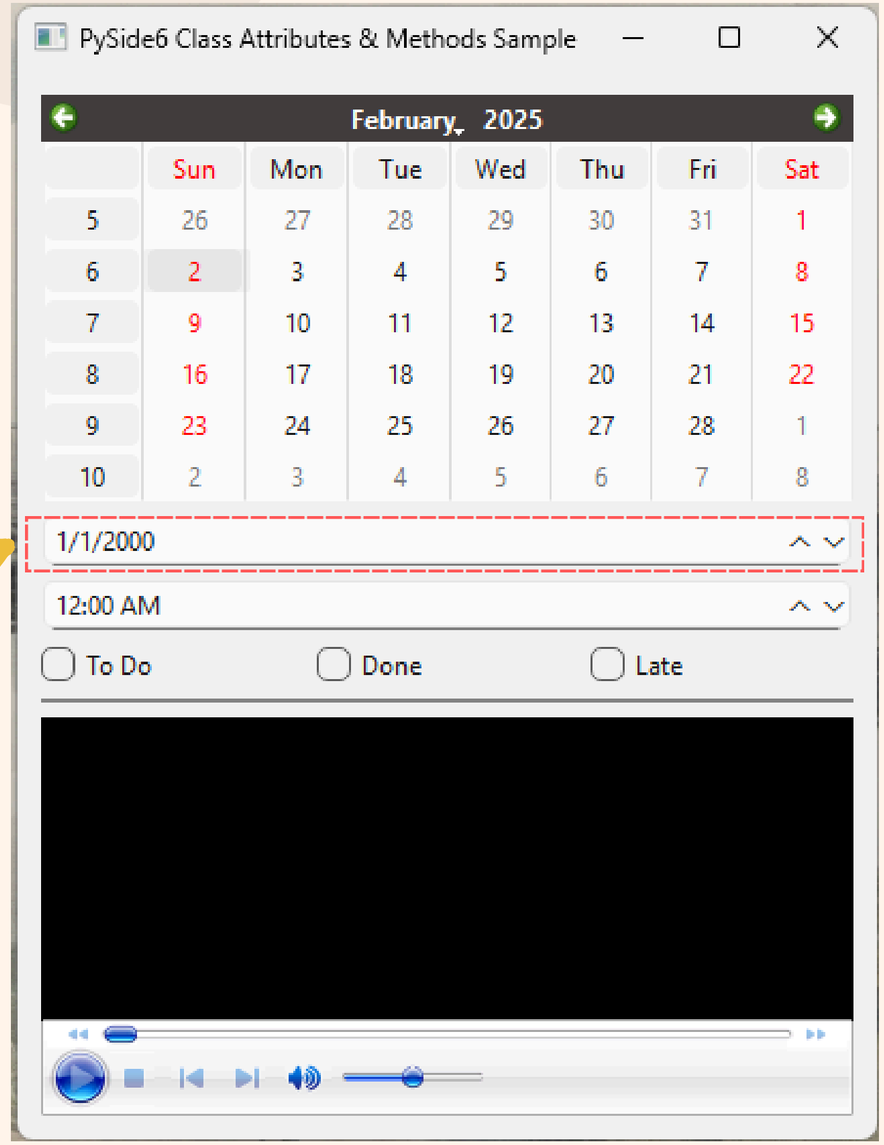
- selectedDate – The currently selected date.



QDate Edit

Methods

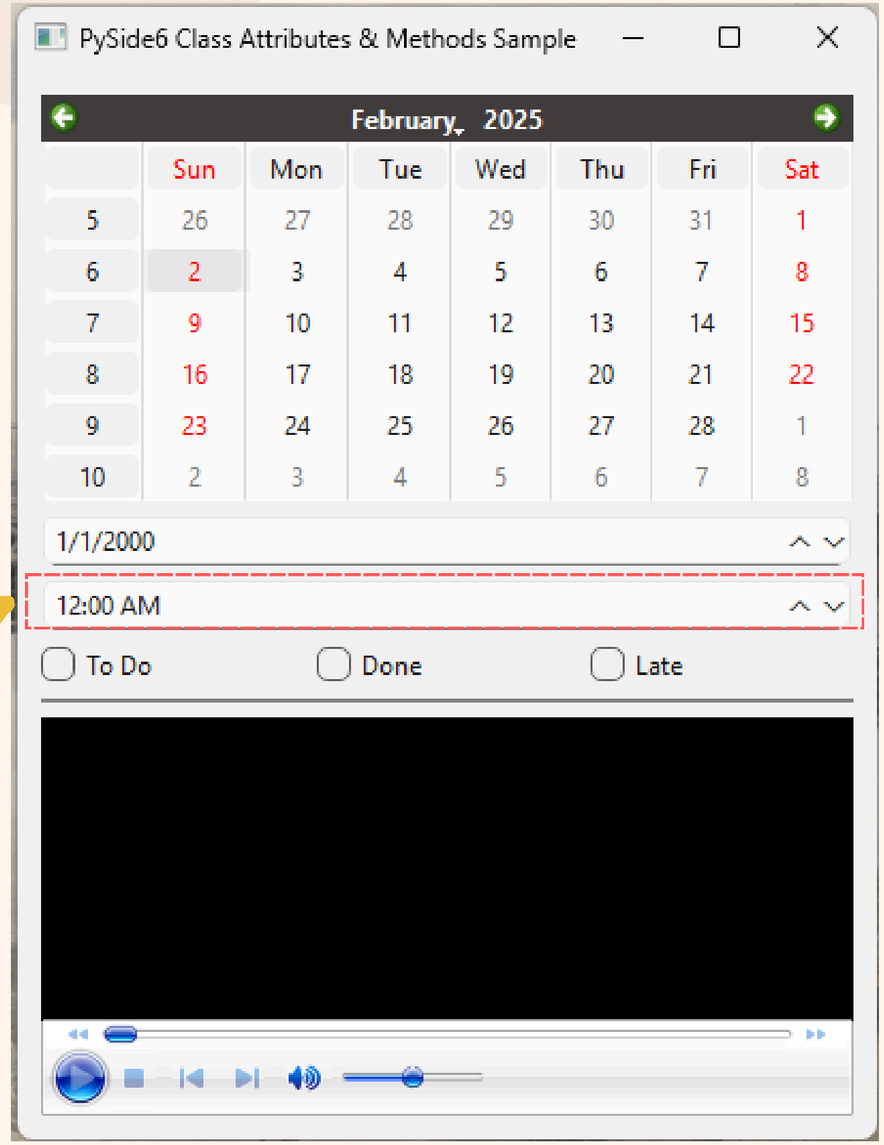
- setDate(QDate) – Sets the date.
- setCalendarPopup(bool) – Enables/disables a popup calendar.
- def __init__()



QDate Edit

Attributes

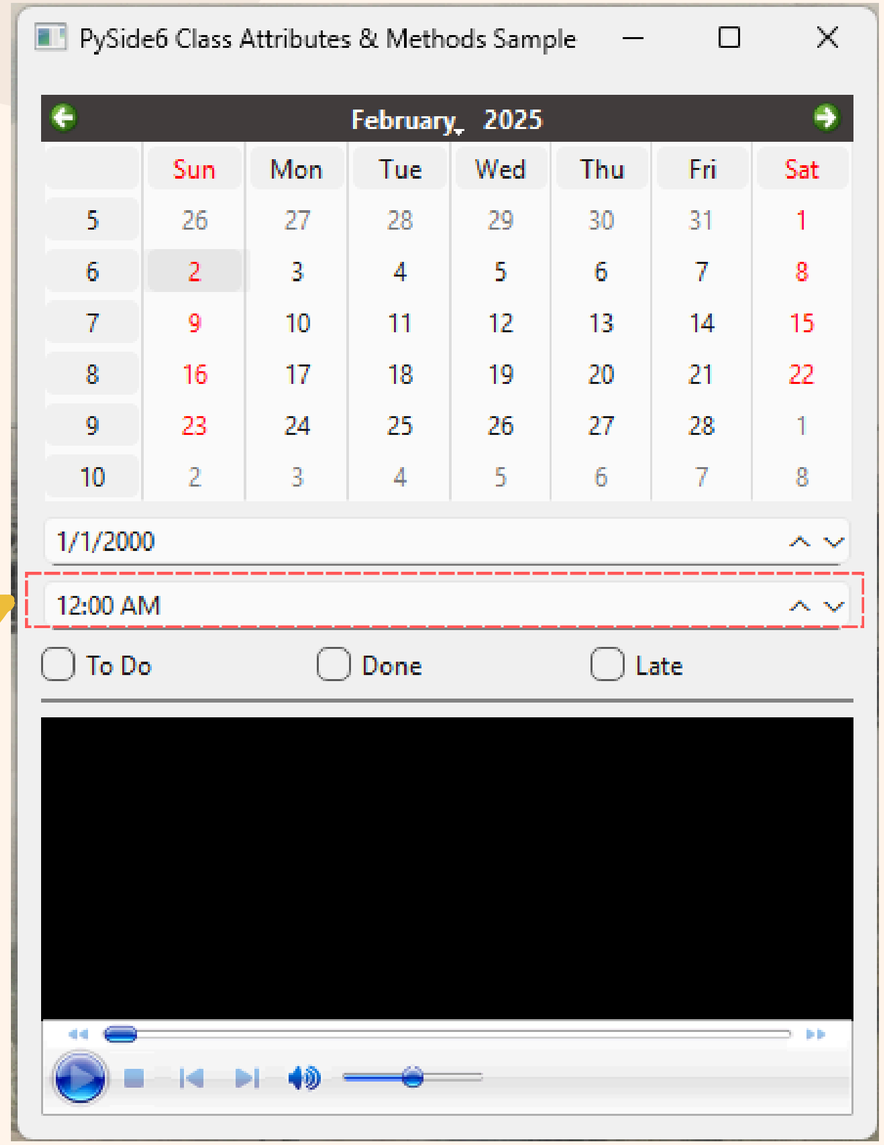
- Date – The current date.



QTime Edit

Methods

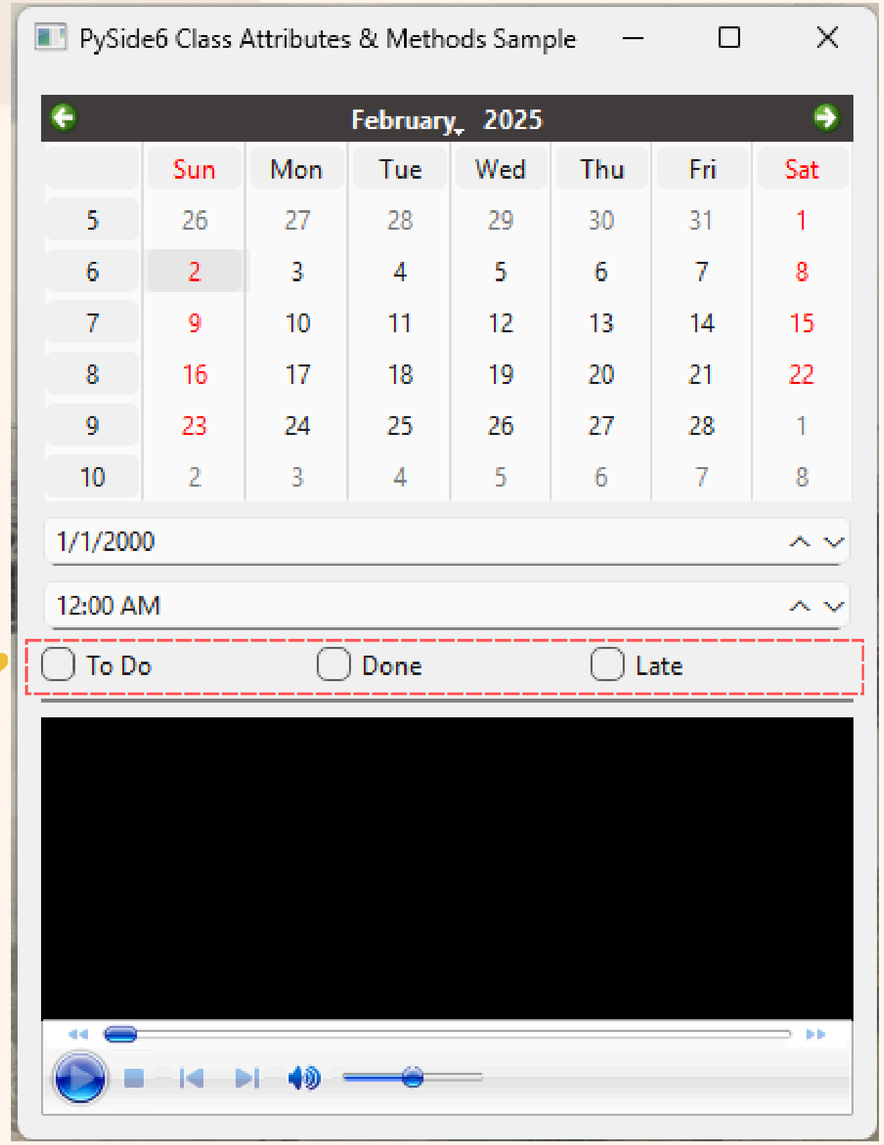
- setTime(QTime) – Sets the time.
- def __init__()
- def __reduce__()
- def __repr__()
- def addMSecs()
- def addSecs()
- def hour()
- def isNull()
- def isValid()
- def minute()
- and more



QTime Edit

Attributes

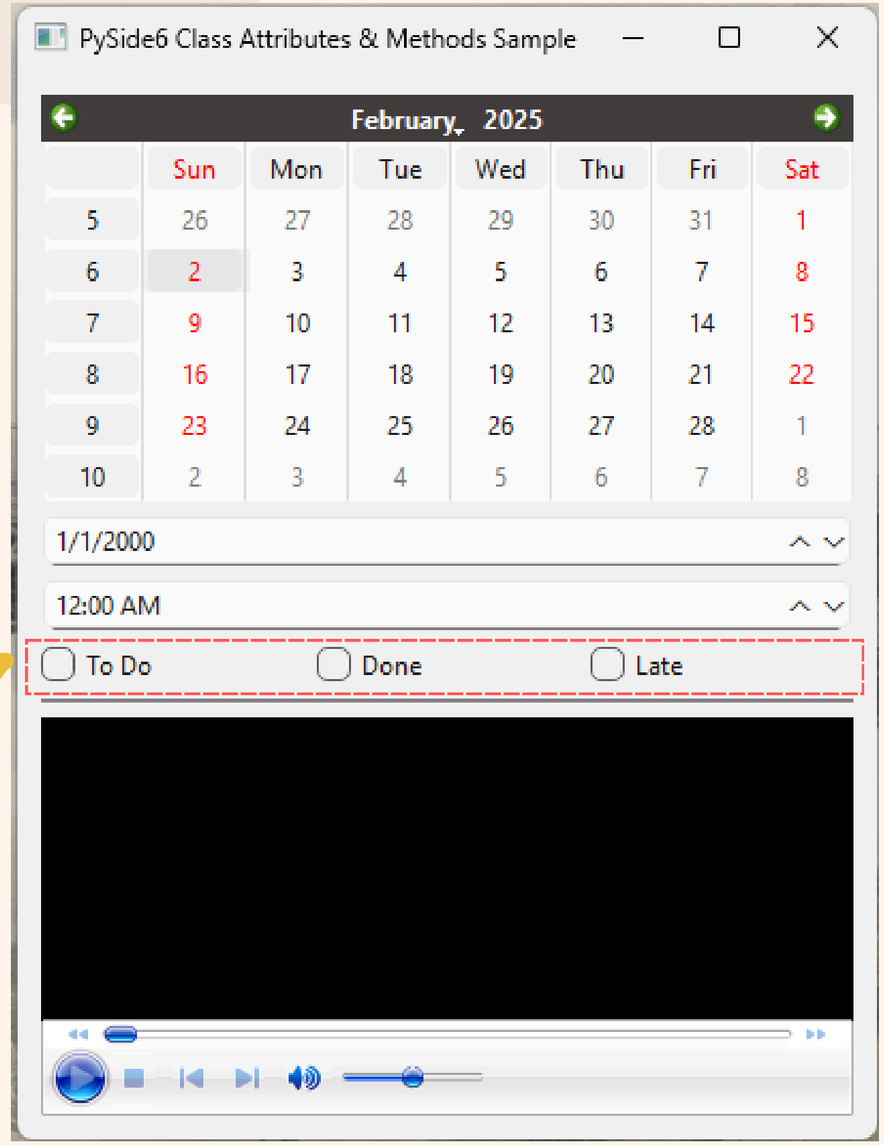
- Time – The currently set time.



QCheck Box

Methods

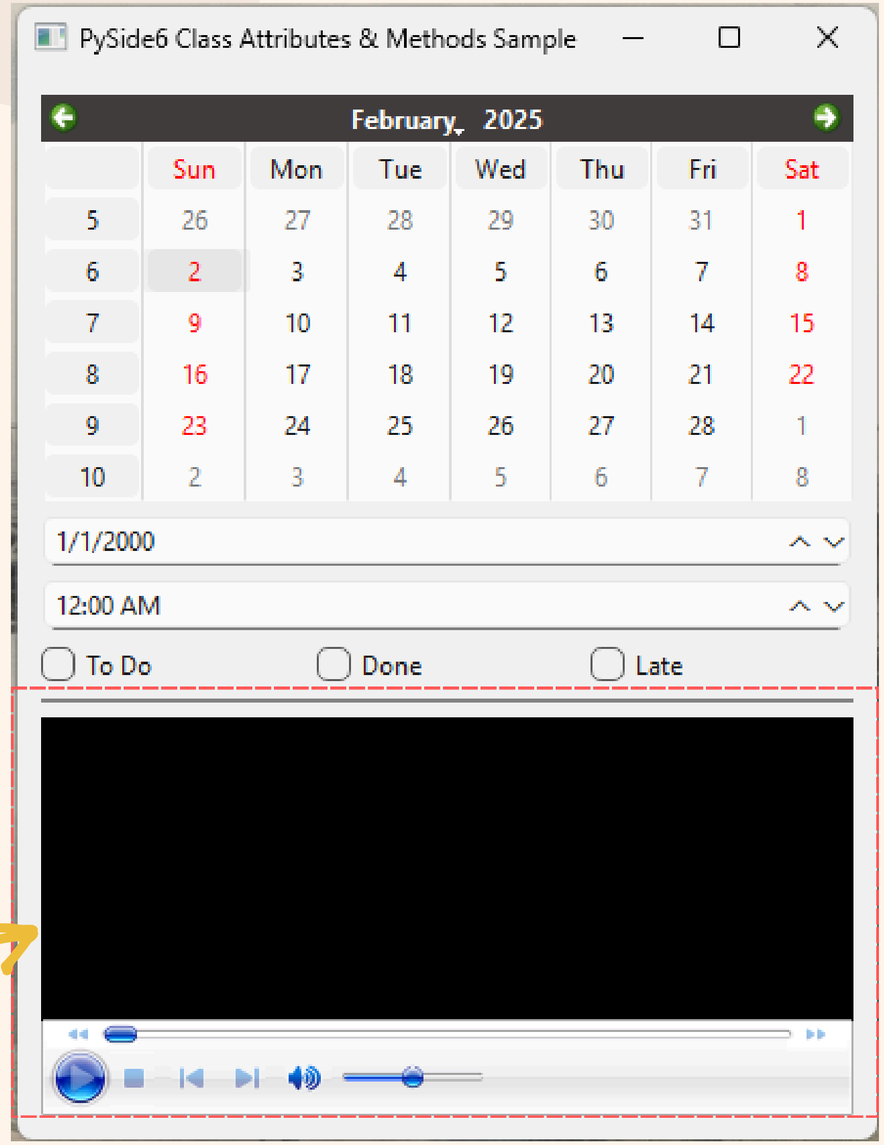
- setChecked(bool) – Sets the checkbox state.
- isChecked() – Returns True if checked.
- def __init__()
- def checkState()
- def isTristate()
- def setCheckState()
- def setTristate()



QCheck Box

Attributes

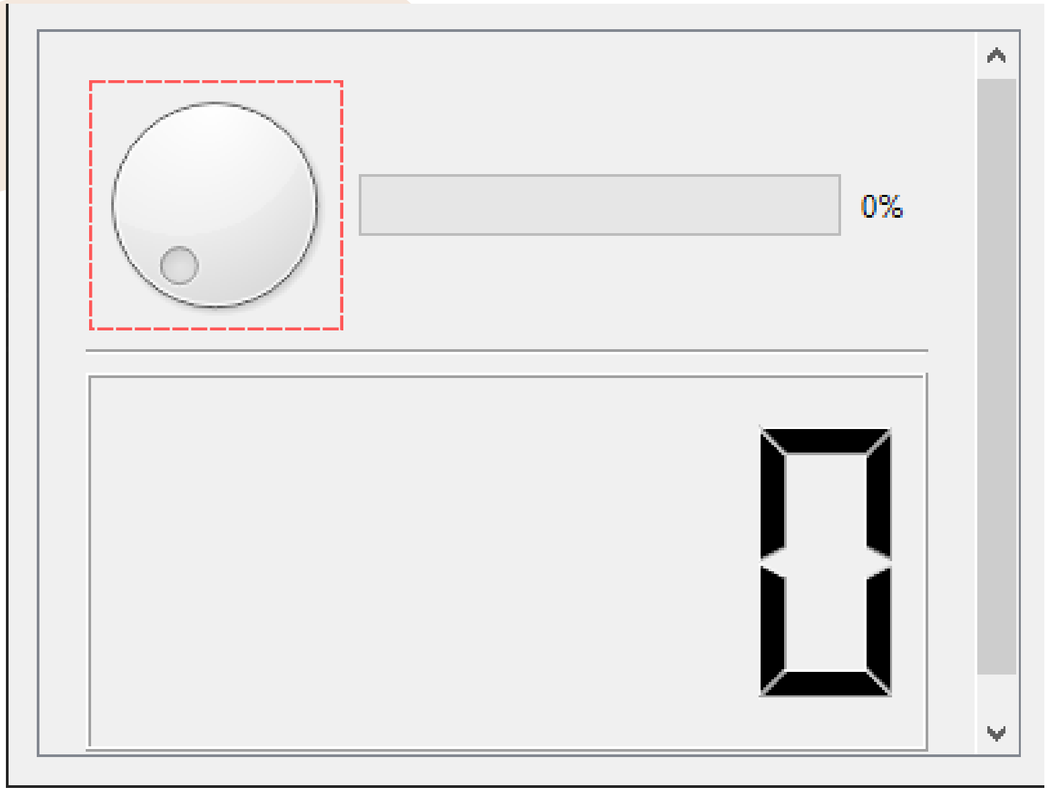
- text – The label text of the checkbox.
- checked – Boolean for whether the checkbox is checked.



QAxWidget

Methods

- setControl(str) – Sets the ActiveX control using a CLSID.

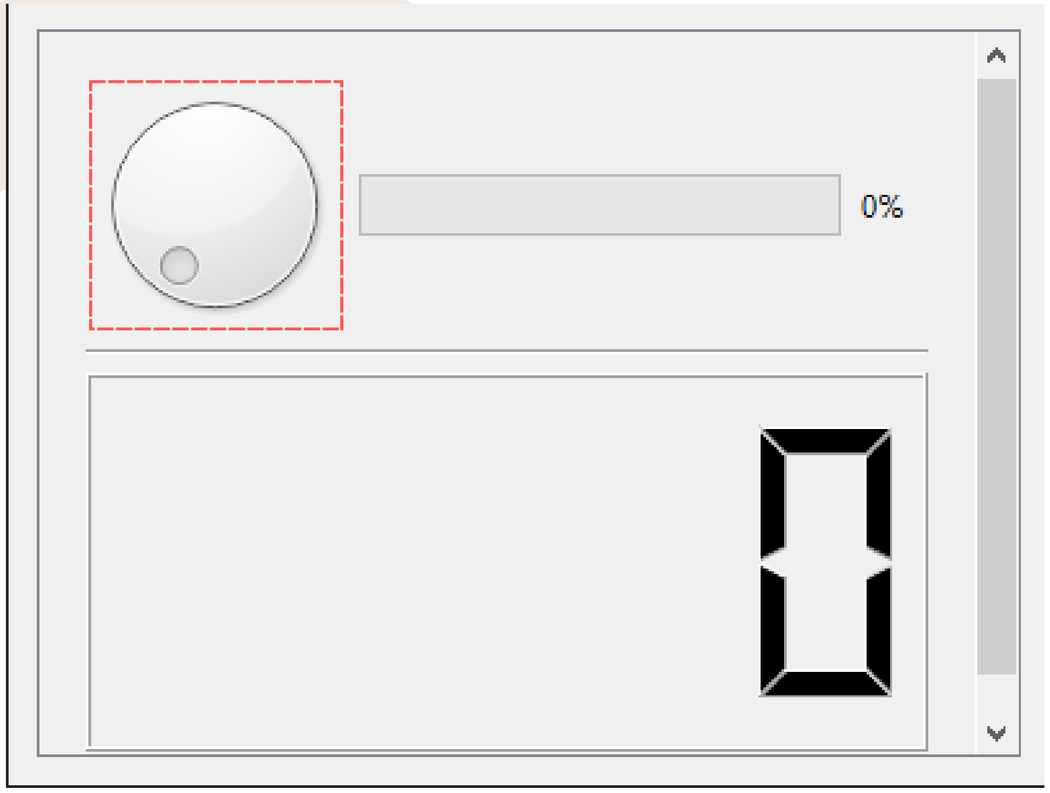


QDial

Method

- value(): Gets the current dial value.
- setValue(int): Sets a new value.
- valueChanged: A signal emitted when the value changes.

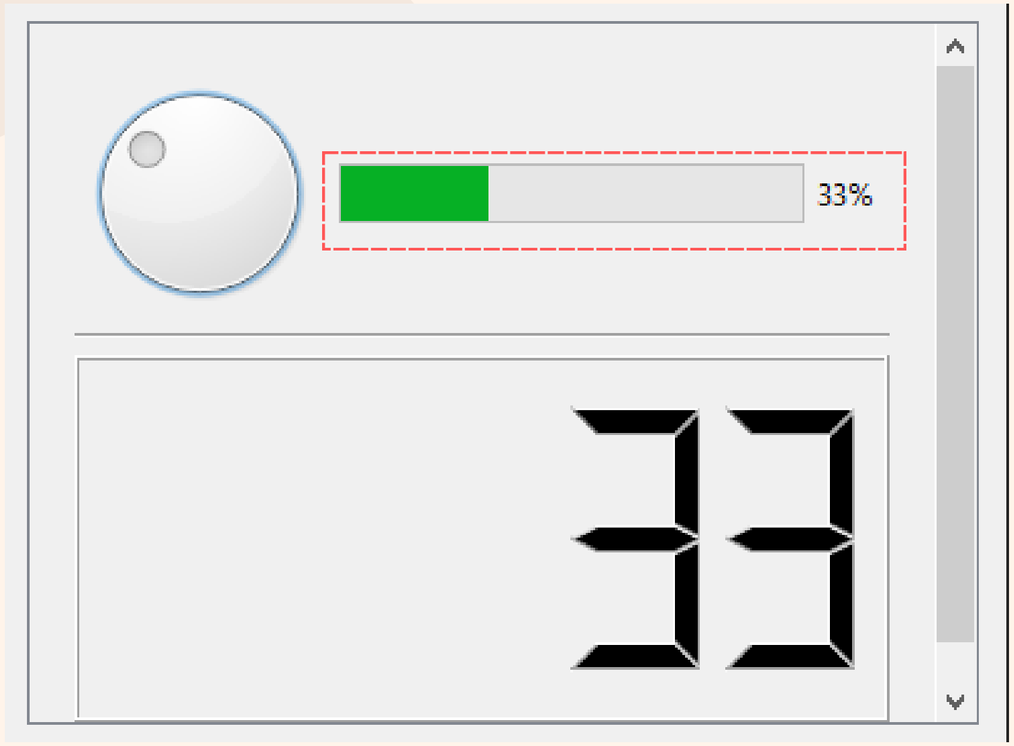
QDial



other methods

- def notchSize()
- def notchTarget()
- def notchesVisible()
- def setNotchTarget()
- def wrapping()

QProgressBar



Method

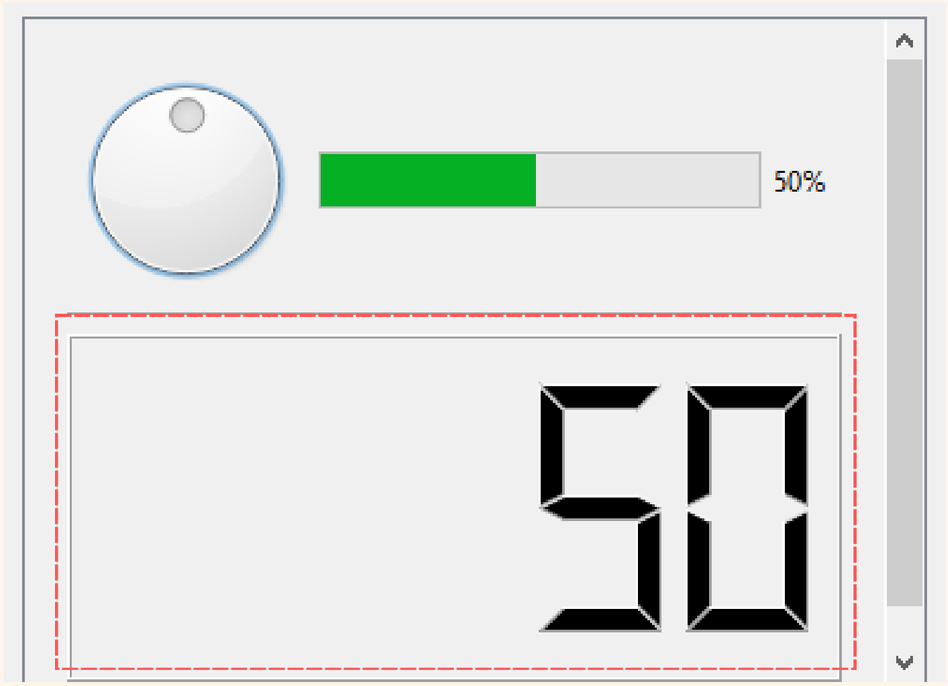
- setValue(int): Sets the progress percentage.
- value(): Retrieves the current progress value.

QProgressBar

Other Method

- def alignment()
- def format()
- def invertedAppearance()
- def isTextVisible()
- def maximum()
- def minimum()
- def orientation()
- def resetFormat()
- def setAlignment()
- def setFormat()
- def setInvertedAppearance()
- def setTextDirection()
- def setTextVisible()
- def textDirection()
- def value()

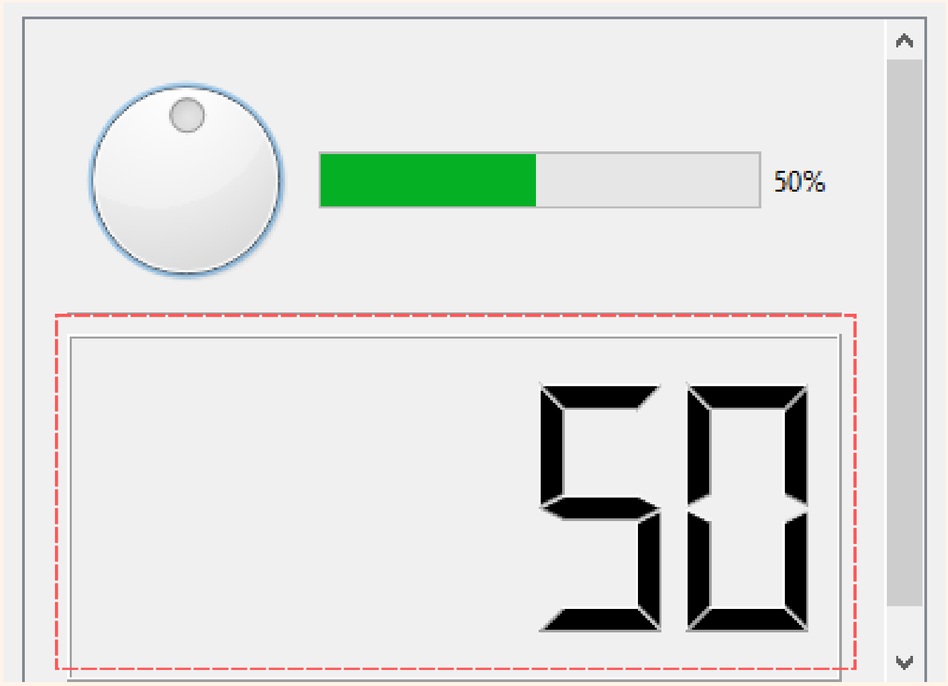
QLCDNumber



Other Methods

- def checkOverflow()
- def digitCount()
- def intValue()
- def mode()
- def segmentStyle()
- def setDigitCount()
- def setMode()
- def setSegmentStyle()
- def smallDecimalPoint()
- def value()

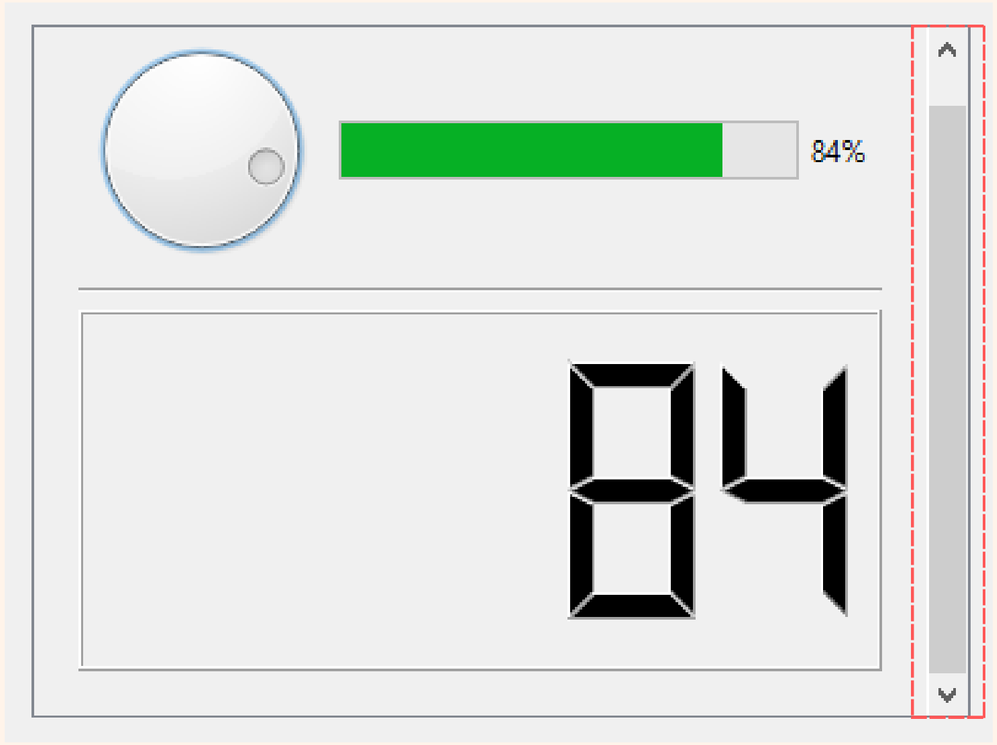
QLCDNumber



Methods

- setValue(int): Sets the progress percentage.
- value(): Retrieves the current progress value.

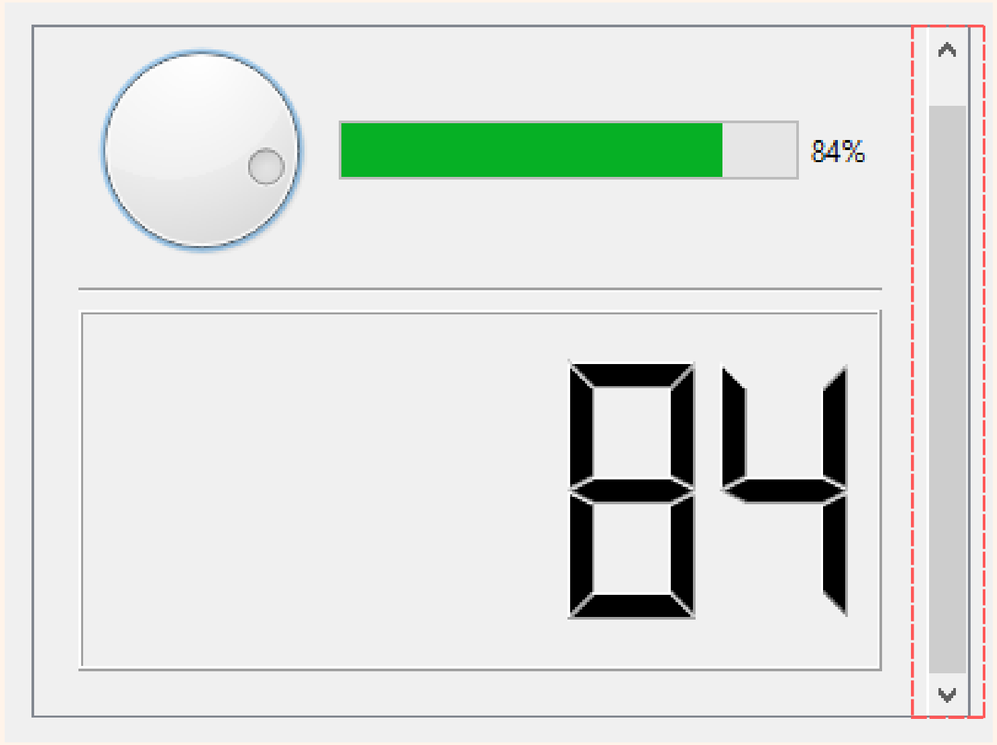
QScrollArea



Methods

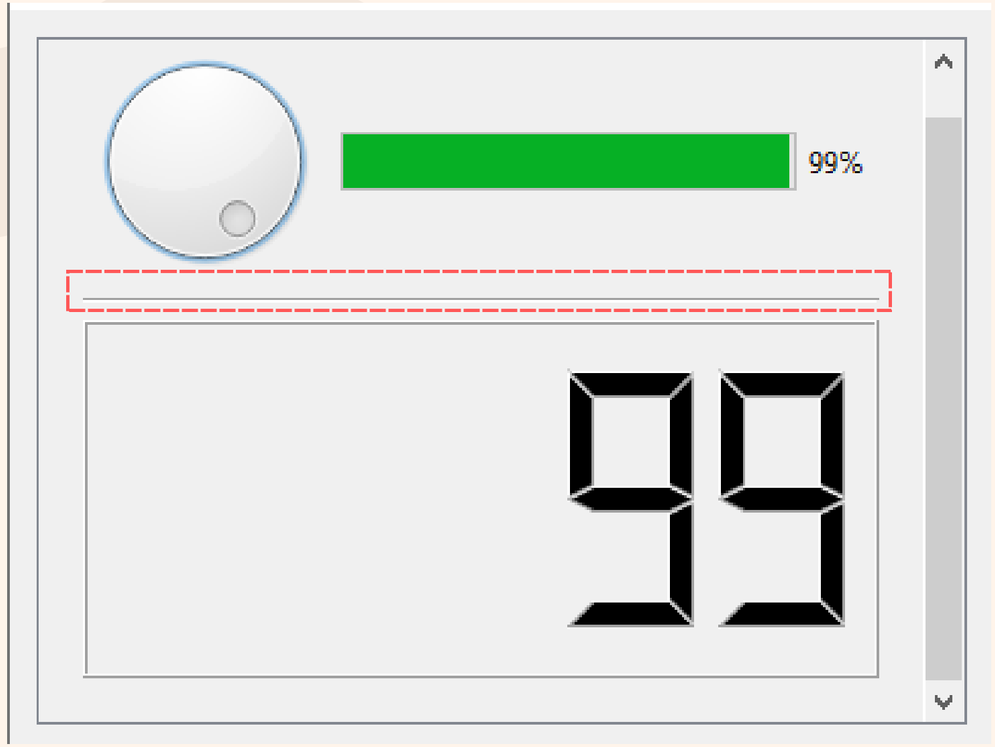
- setValue(int): Sets the progress percentage.
- value(): Retrieves the current progress value.

QScrollArea



Other Methods

- def alignment()
- def ensureVisible()
- def ensureWidgetVisible()
- def setAlignment()
- def setWidget()
- def setWidgetResizable()
- def takeWidget()
- def widget()
- def widgetResizable()



QVBoxLayout

Method

- addWidget(widget): Adds a widget to the vertical layout.
- setSpacing(int): Sets spacing between widgets.

QVBoxLayout

Other Method

- def addLayout()
- def addSpacerItem()
- def addSpacing()
- def addStretch()
- def addStrut()
- def addWidget()
- def direction()
- def insertItem()
- def insertLayout()
- def insertSpacerItem()
- def insertSpacing()
- def insertStretch()
- def insertWidget()
- def setDirection()
- def setStretch()
- def setStretchFactor()
- def stretch()

THANK YOU!