COMPSYS302: PyQt5

2. Widget

Ho Seok AHN

hs.ahn@auckland.ac.nz

THE UNIVERSITY OF AUCKLAND
FACULTY OF ENGINEERING

2. Widget

□ Widget

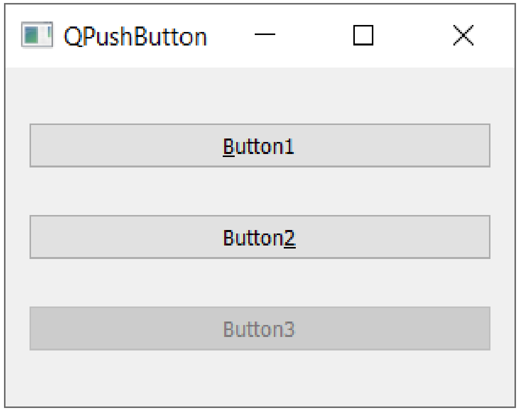
- PyQt supports different widgets such as
  - QPushButton
  - QLabel
  - QCheckBox
  - QRadioButton
  - QComboBox
  - QLineEdit
  - QProgressBar
  - QSlider, QDial
  - QSplitter
  - QGroupBox
  - QTabWidget
  - QPixmap
  - QCalendarWidget
  - QSpinBox
  - QDoubleSpinBox
  - QDateEdit
  - QTimeEdit
  - QDateTimeEdit
  - QTextBrowser
  - QTextEdit

2. Widget

- **Widget**
  - Methods and signals
    - setCheckable(): Set the widget’s status can be kept if it is True
    - toggle(): Change its status
    - setIcon(): Set icon
    - setEnabled(): Cannot use the widget if it is False
    - isChecked(): Return the widget is activated or not
    - setText(): Set the text on the widget
    - text(): Return the text on the widget
    - checkState(): Return its status
    - clicked(): Signal when the widget is clicked
    - pressed(): Signal when the widget is pressed
    - released(): Signal when the widget is released
    - toggled(): Signal when the widget’s status is changed

2.1 QPushButton

□ QPushButton



- Ex) 2_01_QPushButton.py

btn1 = **QPushButton**(text, class)

btn1.**setCheckable**(True)

btn1.**toggle**()

➔ QPushButton(): set text and its class to be assigned.

➔ setCheckable(True): set the button's status can be kept.

➔ toggle(): change its status.

btn2 = **QPushButton**(self)

btn2.**setText**('Button&2')

➔ Can set the text later.

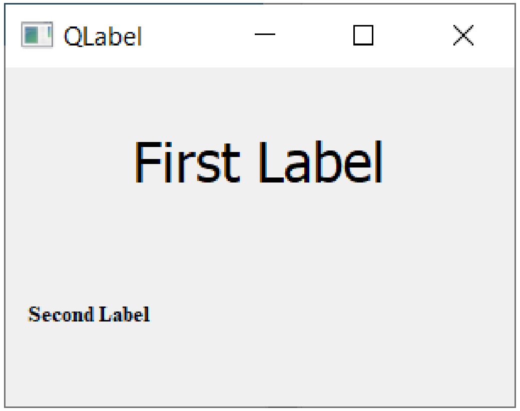
btn3.**setEnabled**(False)

➔ Can make the button unable to use.

2.2 QLabel

□ QLabel

- Set the text or image label.
- No action related to.



First Label

Second Label

  - Ex) 2_02_QLabel.py

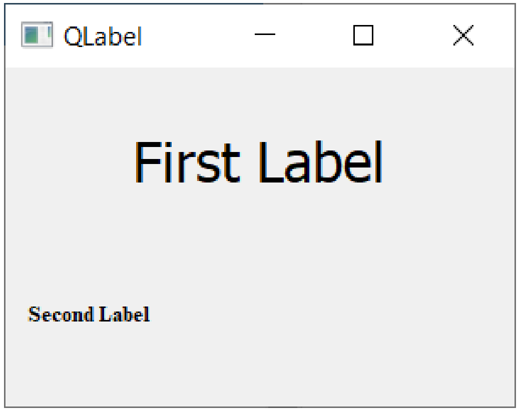
```
label1 = QLabel('First Label', self)
label1.setAlignment(Qt.AlignCenter)
```

➔ Set its text and align its position as centre.

➔ Qt.**AlignVCenter** for aligning vertically only.
Qt.**AlignHCenter** for aligning horizontally only.

2.2 QLabel

□ QLabel



First Label

Second Label

- Ex) 2_02_QLabel.py

```
font1 = label1.font()
font1.setPointSize(20)
```

➔ Set its font size as 20.

➔ Its font style is set as default: MS UI Gothic for Win7

```
font2 = label2.font()
font2.setFamily('Times New Roman')
font2.setBold(True)
```

➔ Set its font style as 'Times New Roman'.

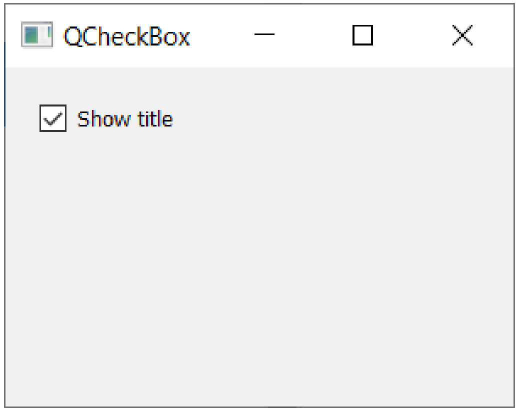
➔ Set its font as bold.

➔ Its font size is set as default: 13.

2.3 QCheckBox

□ QCheckBox

- Default checkbox has two states: on / off.
- Can have three states (inc. 'no change') with setTristate().



QCheckBox

☑ Show title

  - Ex) 2_03_QCheckBox.py

```
cb = QCheckBox('Show title', self)
```

➔ Make a checkbox with 'Show title' as its text.

```
cb.toggle()
```

➔ The default state is 'off', so it makes the state as 'on'.

```
cb.stateChanged.connect(self.changeTitle)
```

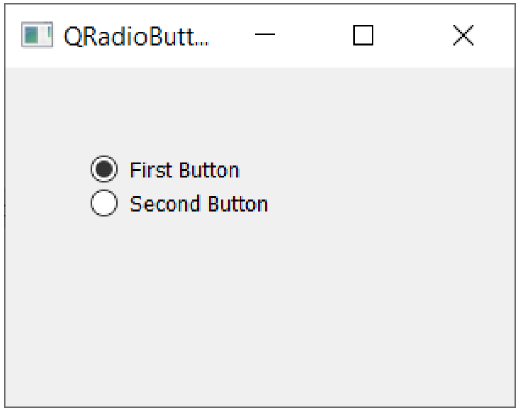
➔ Call the changeTitle() when its state is changed.

2.4 QRadioButton

THE UNIVERSITY OF AUCKLAND
FACULTY OF ENGINEERING

□ QRadioButton

- Select only one of the options in the same group.
- Can have multiple options with setAutoExclusive(False).



- Ex) 2_04_QRadioButton.py

```
rbtn1 = QRadioButton('First Button', self)
```

➔ Make a radiobutton with 'First Button' as its text.

```
rbtn1.setChecked(True)
```

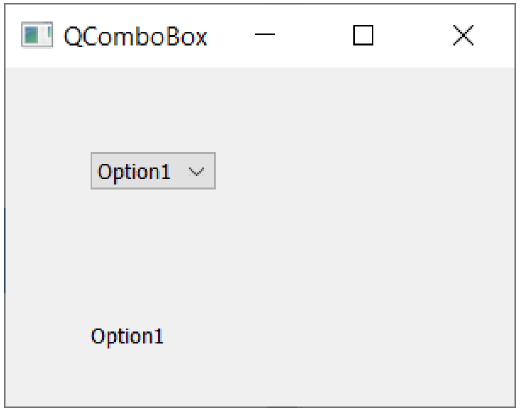
➔ Select this radiobutton as default.

➔ If you call setChecked(True) for the next radiobutton, the last one is selected.

2.5 QComboBox

□ QComboBox

- Select only one of the options in the box.
- Can use it when you have limited space.



- Ex) 2_05_QComboBox.py

```
cb = QComboBox(self)
cb.addItem('Option1')
cb.addItem('Option2')
cb.addItem('Option3')
cb.addItem('Option4')
```

→ Make a combobox with 4 options.

```
cb.activated[str].connect(self.onActivated)
```

→ Call the onActivated() when its option is changed.

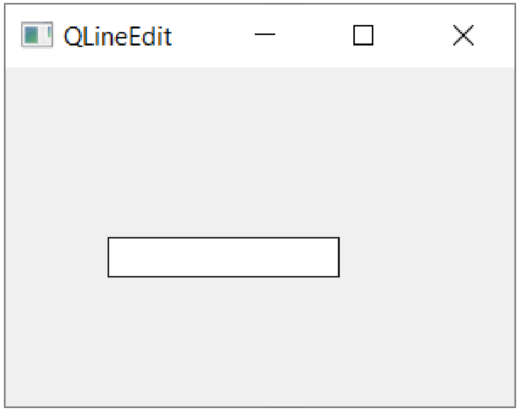
→ Give its text as a parameter.

2.6 QLineEdit

□ QLineEdit

- Widget for one line string.
- Can have different types of string options.

  See https://doc.qt.io/qt-5/qlineedit.html



  - Ex) 2_06_QLineEdit.py

    qle = **QLineEdit**(self)

    ➔ Make a line edit.

    qle.**textChanged**[str].connect(self.onChanged)

    ➔ Call the onChanged() when its text is changed.

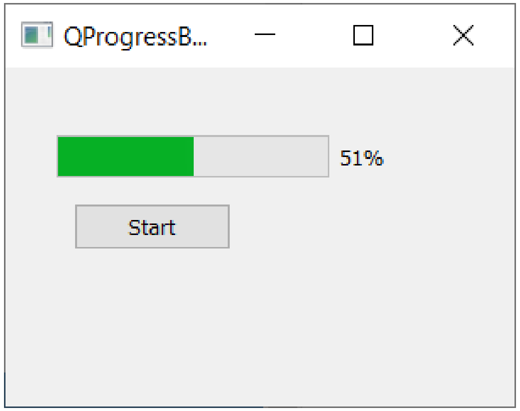
    ➔ Give its text as a parameter.

2.7 QProgressBar

□ QProgressBar

- Use it to show the required time for its process.
- Can have different options.

  See https://doc.qt.io/qt-5/qprogressbar.html



  - Ex) 2_07_QProgressBar.py

    self.pbar = **QProgressBar**(self)

    ➔ Make a progressbar.

    self.pbar.**setValue**(self.step)

    ➔ Update the progressbar.

    self.timer = **QBasicTimer**()

    self.**timer**.start(100, self)

    ➔ Start a timer.

2.8 QDial & QSlider

□ QDial

- Can have different options.

  See https://doc.qt.io/qt-5/qdial.html

  - Ex) 2_08_QSliderQDial.py

    self.dial = **QDial**(self)

    ➔ Make a dial.

    self.dial.**setRange**(0, 50)

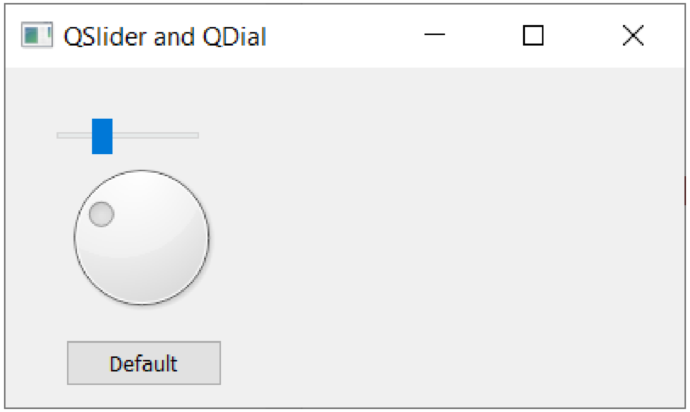
    ➔ Set the range and the step size.

    self.dial.**setNotchesVisible**(True)

    ➔ Show notches.

    self.dial.**valueChanged**.connect(self.slider.setValue)

    ➔ Update slider’s value.



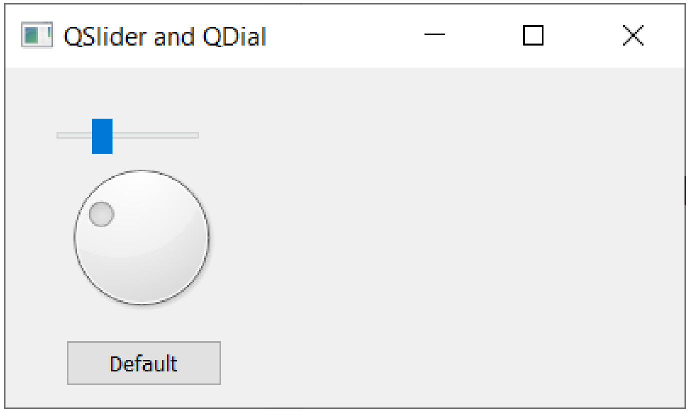
2.8 QSlider & QDial

□ QSlider

- Can have different options.

  See https://doc.qt.io/qt-5/qslider.html

  - Ex) 2_08_QSliderQDial.py



```
self.slider = QSlider(Qt.Horizontal, self)
```

➔ Make a horizontal slider. (Qt.Vertical: vertical slider)

```
self.slider.setRange(0, 50)
self.slider.setSingleStep(2)
```

➔ Set the range and the step size.

```
self.slider.setTickInterval(10)          # Tick interval
self.slider.setTickPosition(2)           # Tick options
```

➔ Set the tick options.

0: no tick / 1: above / 2: below / 3: both sides

2.9 QSplitter

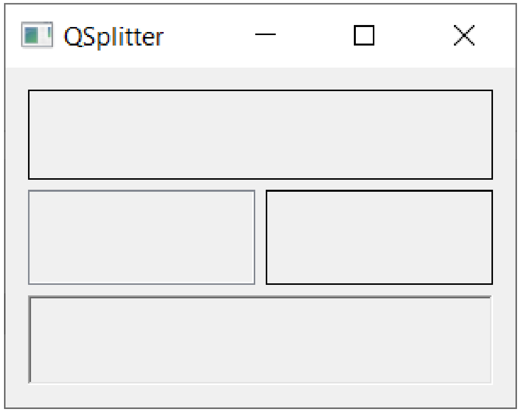
□ QSplitter

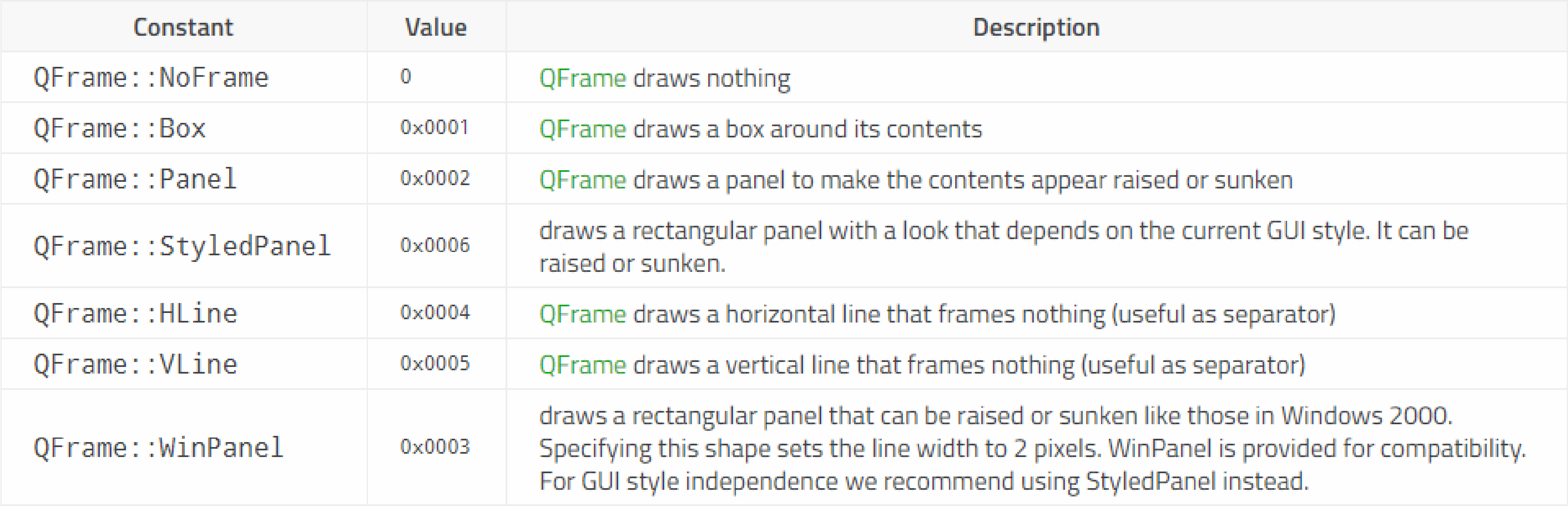
- Adjust the size of frames.
  - Ex) 2_09_QSplitter.py

top = **QFrame**()

top.**setFrameShape**(QFrame.**Box**)

➔ Make a 'top' frame with 'Box' style.



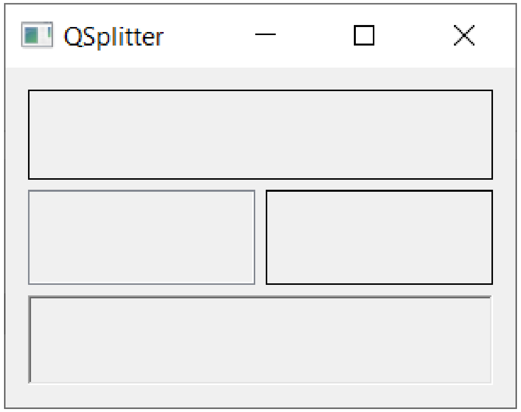


Constant	Value	Description
QFrame::NoFrame	0	QFrame draws nothing
QFrame::Box	0x0001	QFrame draws a box around its contents
QFrame::Panel	0x0002	QFrame draws a panel to make the contents appear raised or sunken
QFrame::StyledPanel	0x0006	draws a rectangular panel with a look that depends on the current GUI style. It can be raised or sunken.
QFrame::HLine	0x0004	QFrame draws a horizontal line that frames nothing (useful as separator)
QFrame::VLine	0x0005	QFrame draws a vertical line that frames nothing (useful as separator)
QFrame::WinPanel	0x0003	draws a rectangular panel that can be raised or sunken like those in Windows 2000. Specifying this shape sets the line width to 2 pixels. WinPanel is provided for compatibility. For GUI style independence we recommend using StyledPanel instead.

2.9 QSplitter

□ QSplitter

- Ex) 2_09_QSplitter.py



```
splitter1 = QSplitter(Qt.Horizontal)
splitter1.addWidget(midleft)
splitter1.addWidget(midright)
```

➔ Make a horizontal splitter, and add two frames first.

```
splitter2 = QSplitter(Qt.Vertical)
splitter2.addWidget(top)
splitter2.addWidget(splitter1)
splitter2.addWidget(bottom)
```

➔ Make a vertical splitter, and add the 1st splitter and two frames.

```
hbox.addWidget(splitter2)
self.setLayout(hbox)
```

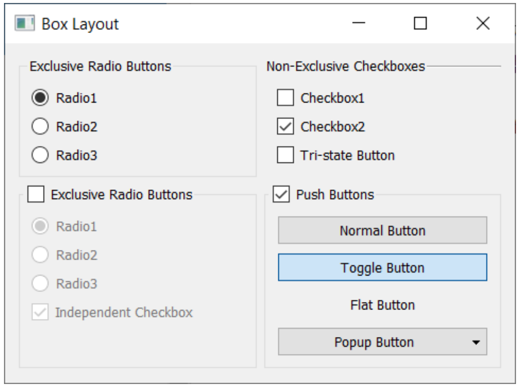
➔ Add the 2nd splitter to a box layout.

2.10 QGroupBox

□ QGroupBox

- Can group widgets.
- See the details.

  https://doc.qt.io/qt-5/qgroupbox.html#details



Box Layout

Exclusive Radio Buttons
● Radio1
○ Radio2
○ Radio3

Non-Exclusive Checkboxes
☐ Checkbox1
☑ Checkbox2
☐ Tri-state Button

☐ Exclusive Radio Buttons
● Radio1
○ Radio2
○ Radio3
☑ Independent Checkbox

☑ Push Buttons

- Ex) 2_10_QGroupBox.py

```
grid = QGridLayout()
grid.addWidget(self.createFirstExclusiveGroup(), 0, 0)
grid.addWidget(self.createSecondExclusiveGroup(), 1, 0)
grid.addWidget(self.createNonExclusiveGroup(), 0, 1)
grid.addWidget(self.createPushButtonGroup(), 1, 1)
self.setLayout(grid)
```

➔ Set 4 groupboxes in different positions of grid layout.

2.10 QGroupBox

□ QGroupBox

- Ex) 2_10_QGroupBox.py

```
groupbox = QGroupBox('Exclusive Radio Buttons')
groupbox.setCheckable(True)    # enable groupbox
groupbox.setChecked(False)     # init groupbox
```

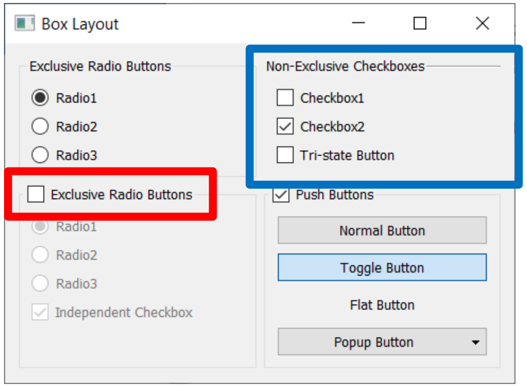
➔ Make a groupbox and set options (RED box).

```
groupbox.setFlat(True)
```

➔ Set different style of groupbox (BLUE box).

```
vbox = QVBoxLayout()
vbox.addWidget(radio1)
vbox.addWidget(radio2)
vbox.addWidget(radio3)
groupbox.setLayout(vbox)
```

➔ Add all the widgets to the box layout.



2.11 QTabWidget

□ QTabWidget

- Can use tabs for caregorizing.
  - Ex) 2_11_QTabWidget.py

```
tab1 = QWidget()
tab2 = QWidget()
```

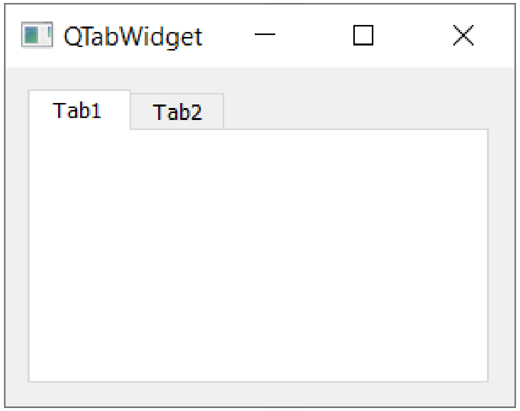
➔ Make two widgets for each tab.

```
tabs = QTabWidget()
tabs.addTab(tab1, 'Tab1')
tabs.addTab(tab2, 'Tab2’)
```

➔ Make tab, and add two widgets.

```
vbox = QVBoxLayout()
vbox.addWidget(tabs)
```

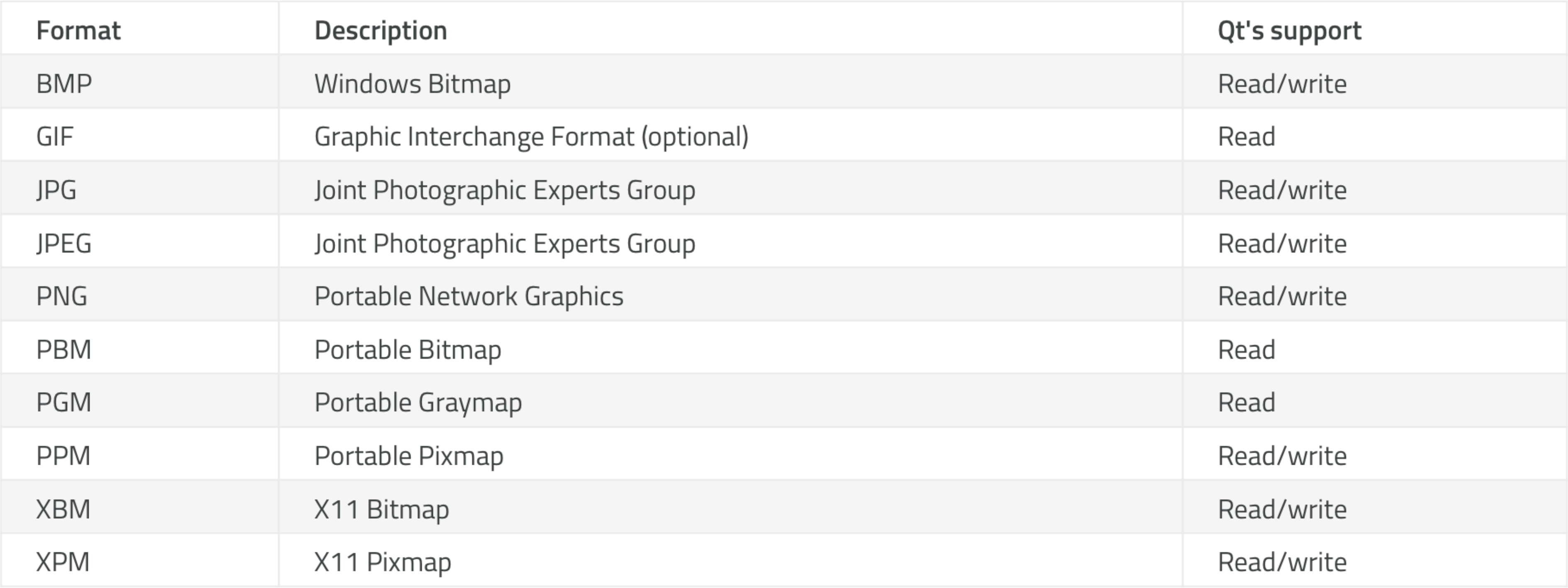
➔ Add the tab to the box layout.



2.12 QPixmap

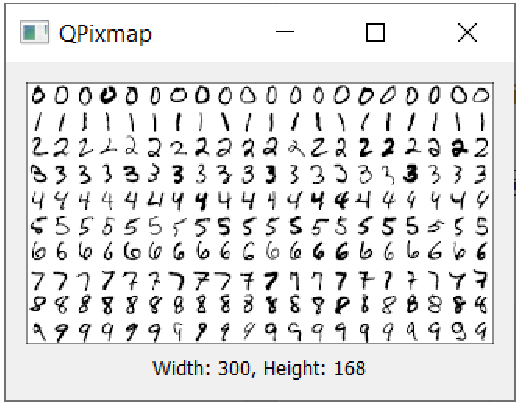
□ QPixmap

- For images.
- It supports following formats.
- Some of them are not writable.



Format	Description	Qt's support
BMP	Windows Bitmap	Read/write
GIF	Graphic Interchange Format (optional)	Read
JPG	Joint Photographic Experts Group	Read/write
JPEG	Joint Photographic Experts Group	Read/write
PNG	Portable Network Graphics	Read/write
PBM	Portable Bitmap	Read
PGM	Portable Graymap	Read
PPM	Portable Pixmap	Read/write
XBM	X11 Bitmap	Read/write
XPM	X11 Pixmap	Read/write

2.12 QPixmap



Width: 300, Height: 168

□ QPixmap

- For images.
  - Ex) 2_12_QPixmap.py

```
pixmap = QPixmap('MNIST.png')
```

➔ Make a QPixmap object pixmap for 'MNIST.png'.

```
lbl_img = QLabel()
lbl_img.setPixmap(pixmap)
```

➔ Set a label for a QPixmap object pixmap.

```
lbl_size = QLabel('Width: '+str(pixmap.width())+', Height: '+str(pixmap.height()))
lbl_size.setAlignment(Qt.AlignCenter)
```

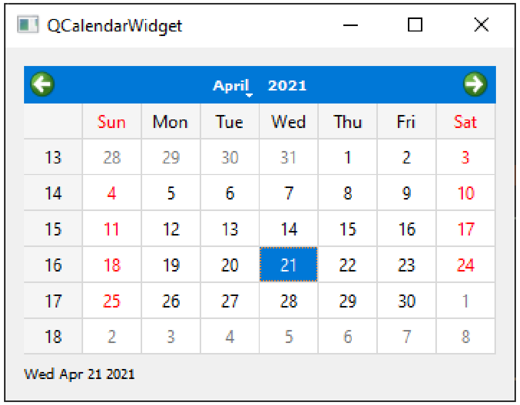
➔ Set another label for showing its width and height.

➔ Show it at the centre.

2.13 QCalendarWidget

□ QCalendarWidget

- For Calendar.
  - Ex) 2_13_QCalendarWidget.py



	Sun	Mon	Tue	Wed	Thu	Fri	Sat
13	28	29	30	31	1	2	3
14	4	5	6	7	8	9	10
15	11	12	13	14	15	16	17
16	18	19	20	21	22	23	24
17	25	26	27	28	29	30	1
18	2	3	4	5	6	7	8

Wed Apr 21 2021

```
cal = QCalendarWidget(self)
cal.setGridVisible(True)
cal.clicked[QDate].connect(self.showDate)
```

→ Make a QCalendarWidget object cal, and show its grid between dates.

→ Call **showDate**() when it is clicked.

```
self.lbl = QLabel(self)
date = cal.selectedDate()
self.lbl.setText(date.toString())
```

→ Set a label for showing the details of the selected date.

→ **selectedDate**() returns the selected date on the calendar. (default: today)

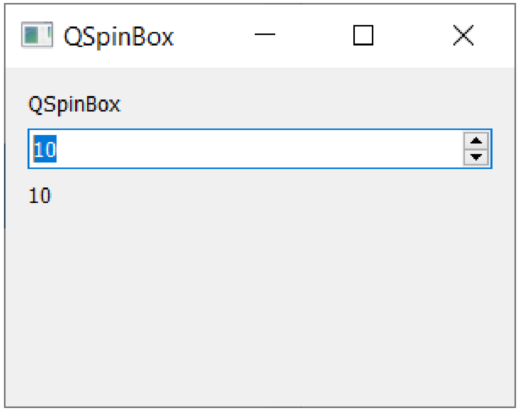
→ Show the returned data.

2.14 QSpinBox

□ QSpinBox

- For integer numbers.

  - Ex) 2_14_QSpinBox.py



QSpinBox

10

10

self.spinbox = **QSpinBox**()

self.spinbox.**setRange**(-10, 30)

➔ Make a spinbox with its range between -10 and 30.

self.spinbox.**setSingleStep**(2)

➔ Set a size of one step as 2.

self.spinbox.**valueChanged**.connect(self.value_changed)

➔ Call value_changed() when its value is changed.

self.lbl2.**setText**(str(self.spinbox.value()))

➔ Update the text with the spinbox value.

2.15 QDoubleSpinBox

□ QDoubleSpinBox

- For float numbers.

  - Ex) 2_15_QDoubleSpinBox.py

```
self.dspinbox = QDoubleSpinBox()
self.dspinbox.setRange(0, 100)
```

→ Make a doublespinbox with its range between 0 and 100.

```
self.dspinbox.setSingleStep(0.5)
```

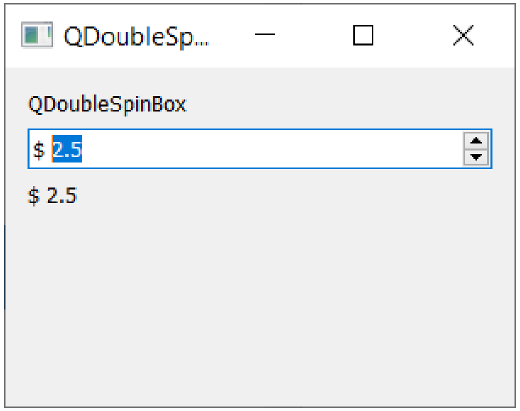
→ Set a size of one step as 0.5.

```
self.dspinbox.setPrefix('$ ')
```

→ Put a character in front of the number.

```
self.dspinbox.setDecimals(1)
```

→ Set the decimal levels.



QDoubleSpinBox

$ 2.5

$ 2.5

2.16 QDateEdit

□ QDateEdit

- For dates.

  - Ex) 2_16_QDateEdit.py

dateedit = **QDateEdit**(self)

➔ Make a dateedit.

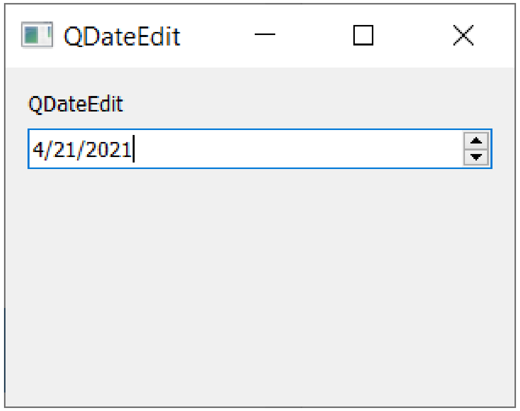
dateedit.**setDate**(QDate.**currentDate**())

➔ Set the date as today using **currentDate**().

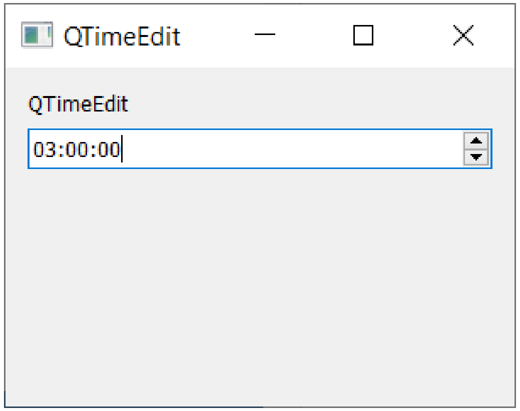
dateedit.**setDateRange**(QDate(1900, 1, 1), QDate(2100, 12, 31))

➔ Set the date range.

➔ Can set min or max of the date using dateedit.**setMinimumDate**() and dateedit.**setMaximumDate**()



2.17 QTimeEdit



03:00:00

□ QTimeEdit

- For time.
  - Ex) 2_17_QTimeEdit.py

timeedit = **QTimeEdit**(self)

➔ Make a timeedit.

timeedit.**setTime**(QTime.**currentTime**())

➔ Set the time as current time using **currentTime**().

timeedit.**setTimeRange**(QTime(3, 00, 00), QTime(23, 30, 00))

➔ Set the time range.

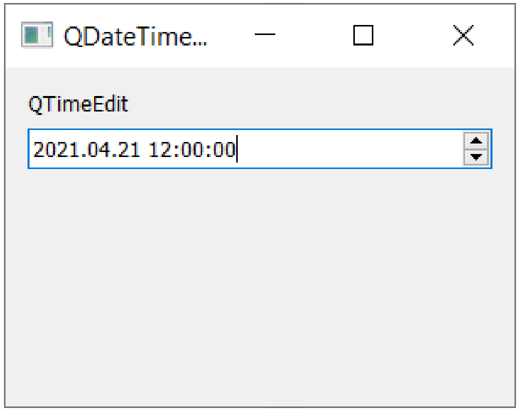
timeedit.**setDisplayFormat**('hh:mm:ss')

➔ Set the time format.

2.18 QDateTimeEdit

□ QDateTimeEdit

- For date and time together.
  - Ex) 2_18_QDateTimeEdit.py



datetimeedit = **QDateTimeEdit**(self)

➔ Make a datetimeedit.

datetimeedit.**setDateTime**(QDateTime.**currentDateTime**())

➔ Set the date and time using **currentDateTime**().

datetimeedit.**setDateTimeRange**(QDateTime(1900, 1, 1, 00, 00, 00), QDateTime(2100, 1, 1, 00, 00, 00))

➔ Set the date and time range.

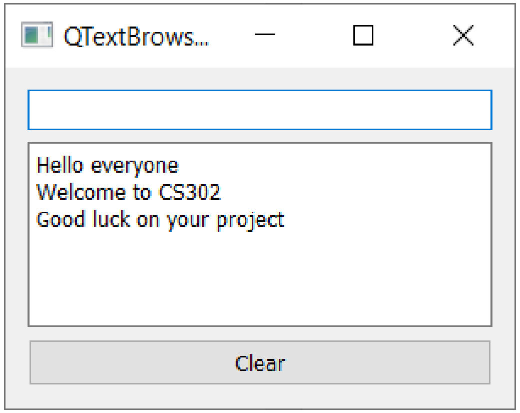
datetimeedit.**setDisplayFormat**('yyyy.MM.dd hh:mm:ss')

➔ Set the date and time format.

2.19 QTextBrowser

□ QTextBrowser

- For multi-line / multi-style text edit, but read only.
  - Ex) 2_19_QTextBrowser.py



Hello everyone
Welcome to CS302
Good luck on your project

```
self.le = QLineEdit()
self.le.returnPressed.connect(self.append_text)
```

➔ Make a lineedit, and connect to append_text() when return key is pressed.

```
self.tb.setAcceptRichText(True)
self.tb.setOpenExternalLinks(True)
```

➔ Can change text style and add external link.

```
text = self.le.text()
self.tb.append(text)
self.le.clear()
```

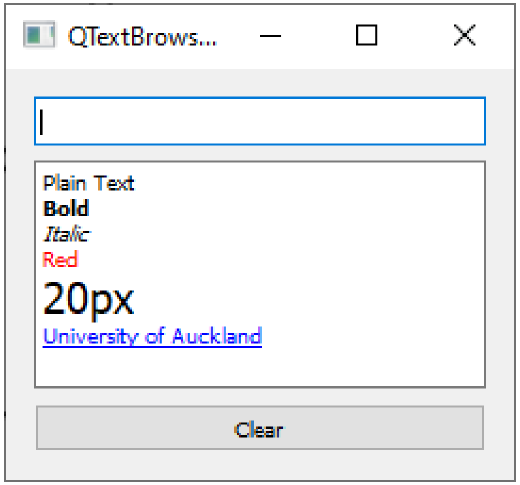
➔ Get text and append it to the text browser, and clear the lineedit.

2.19 QTextBrowser

THE UNIVERSITY OF AUCKLAND
FACULTY OF ENGINEERING

□ QTextBrowser

- Can give different styles.
  - Ex) 2_19_QTextBrowser.py



Plain Text
Bold
Italic
Red
20px
University of Auckland

<b>Bold</b> or <i>Italic</i>

➔ Change the style of text.

<p style="color: red">Red</p>

➔ Change the colour.

<p style="font-size: 20px">20px</p>

➔ Change the font size.

<a href="https://www.auckland.ac.nz">University of Auckland</a>

➔ Add the external link.

2.20 QTextEdit

THE UNIVERSITY OF AUCKLAND
FACULTY OF ENGINEERING

□ QTextEdit

- For multi-line / multi-style text edit, writable as well.
  - Ex) 2_20_QTextEdit.py

```
self.te = QTextEdit()
self.te.setAcceptRichText(False)
```

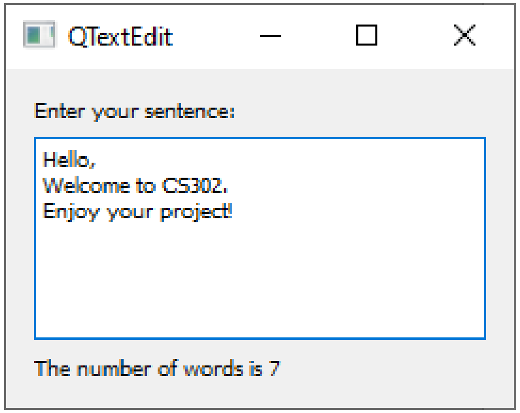
➔ Make a textedit, and disable its style changes.

```
self.te.textChanged.connect(self.text_changed)
```

➔ Call text_changed() when its text is changed.

```
text = self.te.toPlainText()
self.lbl2.setText('The number of words is ' + str(len(text.split())))
```

➔ Get the text, and count the number of words.



Enter your sentence:

Hello,
Welcome to CS302.
Enjoy your project!

The number of words is 7