# pyqtribbon

***Release 0.7.4.post4+gf413afe***

**WANG Hailin**

**Nov 11, 2023**

# CONTENTS:


**1 Getting Started** **3**
- 1.1 Installation . . . 3

**2 The Ribbon Bar** **5**
- 2.1 Introduction . . . 5
- 2.2 Definitions of Ribbon Elements . . . 6
- 2.3 Ribbon Elements in PyQtRibbon . . . 6

**3 User Manual** **7**
- 3.1 The RibbonScreenShotWindow Class . . . 7
- 3.2 Instantiate a Ribbon Bar . . . 7
- 3.3 Customize Ribbon Bar . . . 8
- 3.4 Customize Categories . . . 11
- 3.5 Customize Panels . . . 13
- 3.6 A Complete Example . . . 16

**4 API References** **19**
- 4.1 pyqtribbon package . . . 19

**5 Indices and tables** **49**

**Python Module Index** **51**

**Index** **53**

Ribbon Bar for PyQt or PySide applications.

- GitHub Repository: github.com/haiiliin/pyqtribbon.
- PyPI: pypi.org/project/pyqtribbon.
- Documentation: pyqribbon.readthedocs.io/en/stable.
- Read the Docs: readthedocs.org/projects/pyqtribbon.

CHAPTER

ONE

# GETTING STARTED

## 1.1 Installation

PyQtRibbon is distribued to PyPI, you can use pip to install it:

```
pip install pyqtribbon
```

You can also install the package from source:

```
pip install git+https://github.com/haiiliin/pyqtribbon.git@main
```

CHAPTER

TWO

# THE RIBBON BAR

## 2.1 Introduction

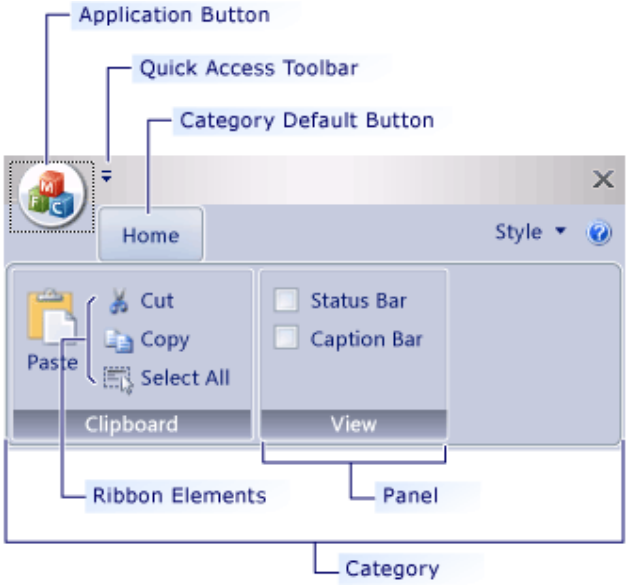
The ribbon is first introduced by Microsoft in the 2000's. It is a toolbar with a tabbed interface. According to Microsoft:

---

**Note:** A ribbon is a user interface (UI) element that organizes commands into logical groups. These groups appear on separate tabs in a strip across the top of the window. The ribbon replaces the menu bar and toolbars. A ribbon can significantly improve application usability. For more information, see Ribbons. The following illustration shows a ribbon. A ribbon can significantly improve application usability. For more information, see Ribbons. The following illustration shows a ribbon.

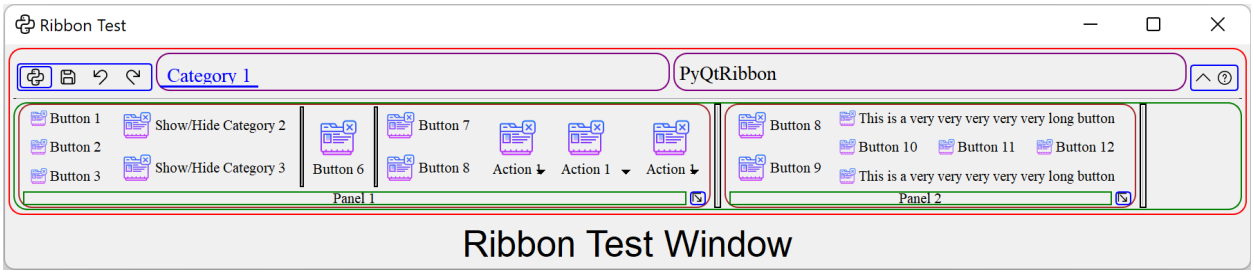

---

## 2.2 Definitions of Ribbon Elements




- **Application button**: The button that appears on the upper-left corner of a ribbon. The Application button replaces the File menu and is visible even when the ribbon is minimized. When the button is clicked, a menu that has a list of commands is displayed.
- **Quick Access toolbar**: A small, customizable toolbar that displays frequently used commands.
- **Category**: The logical grouping that represents the contents of a ribbon tab.
- **Category Default button**: The button that appears on the ribbon when the ribbon is minimized. When the button is clicked, the category reappears as a menu.
- **Panel**: An area of the ribbon bar that displays a group of related controls. Every ribbon category contains one or more ribbon panels.
- **Ribbon elements**: Controls in the panels, for example, buttons and combo boxes. To see the various controls that can be hosted on a ribbon, see RibbonGadgets Sample: Ribbon Gadgets Application.

## 2.3 Ribbon Elements in PyQtRibbon



CHAPTER

# THREE

# USER MANUAL

## 3.1 The RibbonScreenShotWindow Class

The `RibbonScreenShotWindow` class is just for taking a screenshot of the window, the window will be closed 0.1s after it is shown. It is just used for documenting the window.

**class** pyqtribbon.screenshotwindow.**RibbonScreenShotWindow**(*fileName: str = 'shot.jpg'*, **args*, ***kwargs*)

This class is just for taking a screenshot of the window, the window will be closed 0.1s after it is shown.

Initialize the class.

**Parameters**
**fileName** – The file name for the screenshot.

**setScreenShotFileName**(*fileName: str*)

Set the file name for the screenshot.

**Parameters**
**fileName** – The file name for the screenshot.

**takeScreenShot**()

Take a screenshot of the window.

## 3.2 Instantiate a Ribbon Bar

`RibbonBar` is inherited from `QMenuBar`, you can use the *setMenuBar* method of `QMainWindow` to set the ribbon bar as the main menu bar.

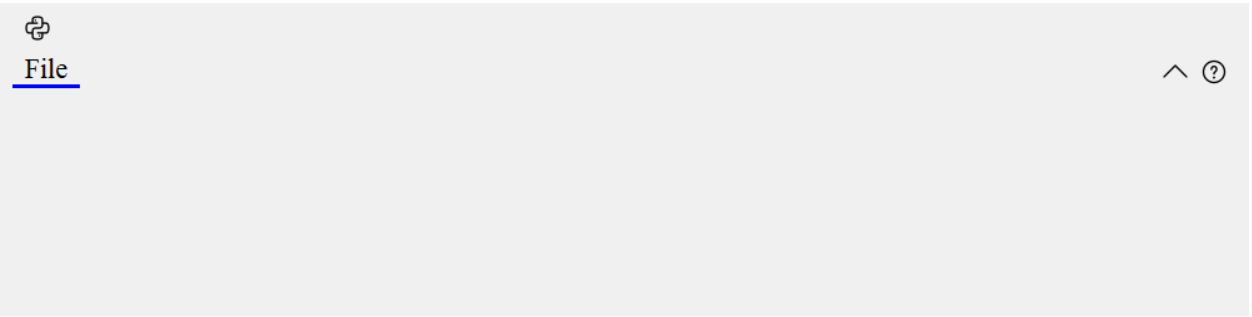
```
from pyqtribbon import RibbonBar

window = QtWidgets.QMainWindow()
ribbon = RibbonBar()
window.setMenuBar(ribbon)
```

### 3.2.1 Example

For example, using the following code,

```
import sys

from qtpy import QtGui, QtWidgets

from pyqtribbon import RibbonBar
from pyqtribbon.screenshotwindow import RibbonScreenShotWindow

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    app.setFont(QtGui.QFont("Times New Roman", 8))
    window = RibbonScreenShotWindow("ribbonbar.png")

    # Ribbon bar
    ribbonbar = RibbonBar()
    window.setMenuBar(ribbonbar)

    # Show the window
    window.resize(1000, 250)
    window.show()

    sys.exit(app.exec_())
```

You can get a window like this:




## 3.3 Customize Ribbon Bar

### 3.3.1 General Setups

| | |
|---|---|
| *RibbonBar.setRibbonStyle*(style) | Set the style of the ribbon. |
| *RibbonBar.ribbonHeight*() | Get the total height of the ribbon. |
| *RibbonBar.setRibbonHeight*(height) | Set the total height of the ribbon. |
| *RibbonBar.showRibbon*() | Show the ribbon. |
| *RibbonBar.hideRibbon*() | Hide the ribbon. |
| *RibbonBar.ribbonVisible*() | Get the visibility of the ribbon. |
| *RibbonBar.setRibbonVisible*(visible) | Set the visibility of the ribbon. |

### 3.3.2 Setup Application Button

| | |
|---|---|
| *RibbonBar.applicationOptionButton*() | Return the application button. |
| *RibbonBar.setApplicationIcon*(icon) | Set the application icon. |
| *RibbonBar.addFileMenu*() | Add a file menu to the ribbon. |

### 3.3.3 Setup Title

| | |
|---|---|
| *RibbonBar.title*() | Return the title of the ribbon. |
| *RibbonBar.setTitle*(title) | Set the title of the ribbon. |
| *RibbonBar.addTitleWidget*(widget) | Add a widget to the title widget. |
| *RibbonBar.insertTitleWidget*(index, widget) | Insert a widget to the title widget. |
| *RibbonBar.removeTitleWidget*(widget) | Remove a widget from the title widget. |

### 3.3.4 Setup Category Tab Bar

| | |
|---|---|
| *RibbonBar.tabBar*() | Return the tab bar of the ribbon. |

### 3.3.5 Setup Quick Access Bar

| | |
|---|---|
| *RibbonBar.quickAccessToolBar*() | Return the quick access toolbar of the ribbon. |
| *RibbonBar.addQuickAccessButton*(button) | Add a button to the quick access bar. |
| *RibbonBar.setQuickAccessButtonHeight*([height]) | Set the height of the quick access buttons. |

### 3.3.6 Setup Right Tool Bar

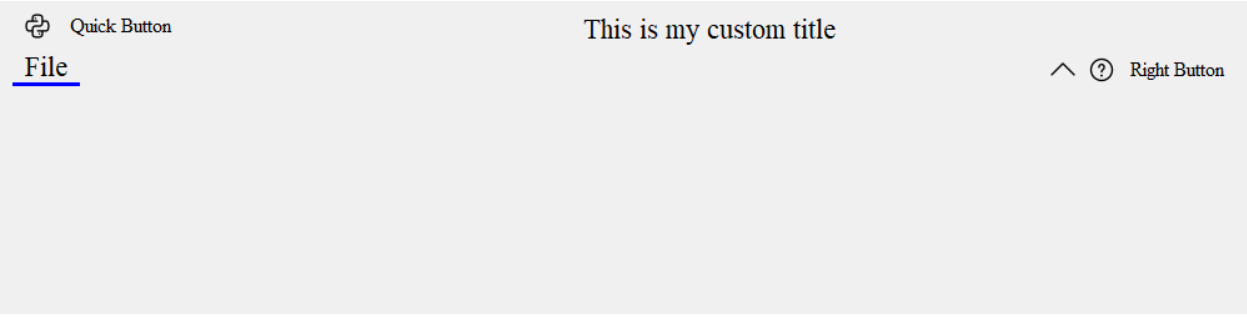
| | |
|---|---|
| *RibbonBar.rightToolBar*() | Return the right toolbar of the ribbon. |
| *RibbonBar.addRightToolButton*(button) | Add a widget to the right button bar. |
| *RibbonBar.setRightToolBarHeight*([height]) | Set the height of the right buttons. |
| *RibbonBar.setHelpButtonIcon*(icon) | Set the icon of the help button. |
| *RibbonBar.removeHelpButton*() | Remove the help button from the ribbon. |
| *RibbonBar.helpButtonClicked*(bool) | Signal, the help button was clicked. |
| *RibbonBar.collapseRibbonButton*() | Return the collapse ribbon button. |
| *RibbonBar.setCollapseButtonIcon*(icon) | Set the icon of the min button. |
| *RibbonBar.removeCollapseButton*() | Remove the min button from the ribbon. |

### 3.3.7 Example

For example, using the following code,

```
import sys

from qtpy import QtGui
from qtpy.QtWidgets import QApplication, QToolButton

from pyqtribbon import RibbonBar
from pyqtribbon.screenshotwindow import RibbonScreenShotWindow

if __name__ == "__main__":
    app = QApplication(sys.argv)
    app.setFont(QtGui.QFont("Times New Roman", 8))
    window = RibbonScreenShotWindow("ribbonbar-customize.png")

    # Ribbon bar
    ribbonbar = RibbonBar()
    window.setMenuBar(ribbonbar)

    # Title of the ribbon
    ribbonbar.setTitle("This is my custom title")

    # Quick Access Bar
    qbutton = QToolButton()
    qbutton.setText("Quick Button")
    ribbonbar.addQuickAccessButton(qbutton)

    # Right toolbar
    rbutton = QToolButton()
    rbutton.setText("Right Button")
    ribbonbar.addRightToolButton(rbutton)

    # Show the window
    window.resize(1000, 250)
    window.show()

    sys.exit(app.exec_())
```

You can get a window like this:



### 3.3.8 Manage Categories

| | |
|---|---|
| *RibbonBar.categories*() | Return a list of categories of the ribbon. |
| *RibbonBar.addCategory*(title[, style, color]) | Add a new category to the ribbon. |
| *RibbonBar.addCategoriesBy*(data) | Add categories from a dict. |
| *RibbonBar.addNormalCategory*(title) | Add a new category to the ribbon. |
| *RibbonBar.addContextCategory*(title[, color]) | Add a new context category to the ribbon. |
| *RibbonBar.addContextCategories*(name, titles) | Add a group of context categories with the same tab color to the ribbon. |
| *RibbonBar.showContextCategory*(category) | Show the given category or categories, if it is not a context category, nothing happens. |
| *RibbonBar.hideContextCategory*(category) | Hide the given category or categories, if it is not a context category, nothing happens. |
| *RibbonBar.removeCategory*(category) | Remove a category from the ribbon. |
| *RibbonBar.setCurrentCategory*(category) | Set the current category. |
| *RibbonBar.currentCategory*() | Return the current category. |
| *RibbonBar.showCategoryByIndex*(index) | Show category by tab index |

## 3.4 Customize Categories

### 3.4.1 Setup Styles

| | |
|---|---|
| *RibbonCategory.categoryStyle*() | Return the button style of the category. |
| *RibbonCategory.setCategoryStyle*(style) | Set the button style of the category. |

### 3.4.2 Manage Panels

| | |
|---|---|
| *RibbonCategory.addPanel*(title[, ...]) | Add a new panel to the category. |
| *RibbonCategory.addPanelsBy*(data) | Add panels from a dictionary. |
| *RibbonCategory.removePanel*(title) | Remove a panel from the category. |
| *RibbonCategory.takePanel*(title) | Remove and return a panel from the category. |
| *RibbonCategory.panel*(title) | Return a panel from the category. |
| *RibbonCategory.panels*() | Return all panels in the category. |

### 3.4.3 Example

For example, using the following code,

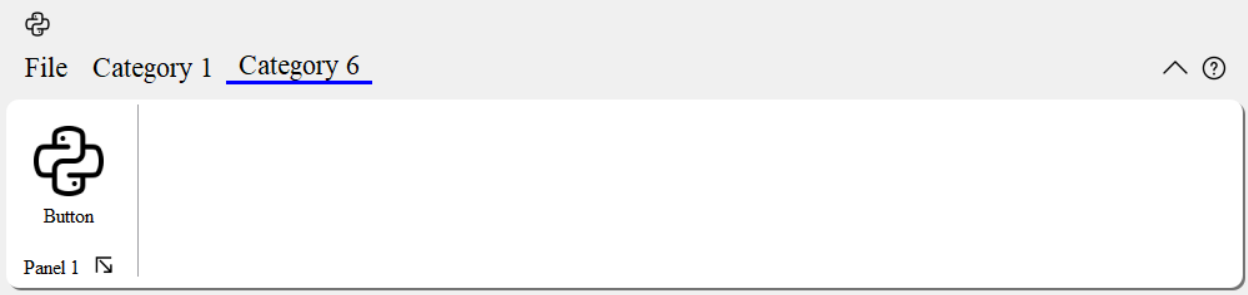
```
import sys

from qtpy import QtGui
from qtpy.QtGui import QIcon
from qtpy.QtWidgets import QApplication

from pyqtribbon import RibbonBar, RibbonCategoryStyle
```

(continues on next page)

(continued from previous page)

```
from pyqtribbon.screenshotwindow import RibbonScreenShotWindow

if __name__ == "__main__":
    app = QApplication(sys.argv)
    app.setFont(QtGui.QFont("Times New Roman", 8))
    window = RibbonScreenShotWindow("category.png")

    # Ribbon bar
    ribbonbar = RibbonBar()
    window.setMenuBar(ribbonbar)

    # Categories
    category1 = ribbonbar.addCategory("Category 1")
    panel1 = category1.addPanel("Panel 1")
    panel1.addLargeButton("Large Button 1", QIcon("python.png"))

    category2 = ribbonbar.addContextCategory("Category 2")
    panel12 = category2.addPanel("Panel 2")
    panel12.addLargeButton("Large Button 2", QIcon("python.png"))

    categories = ribbonbar.addCategoriesBy(
        {
            "Category 6": {
                "style": RibbonCategoryStyle.Normal,
                "panels": {
                    "Panel 1": {
                        "showPanelOptionButton": True,
                        "widgets": {
                            "Button 1": {
                                "type": "Button",
                                "arguments": {
                                    "icon": QIcon("python.png"),
                                    "text": "Button",
                                    "tooltip": "This is a tooltip",
                                },
                            },
                        },
                    },
                },
            }
        }
    )
    ribbonbar.setCurrentCategory(categories["Category 6"])

    # Show the window
    window.resize(1000, 250)
    window.show()

    sys.exit(app.exec_())
```

You can get a window like this:




## 3.5 Customize Panels

### 3.5.1 Setup Title Label

| | |
|---|---|
| *RibbonPanel.title*() | Get the title of the panel. |
| *RibbonPanel.setTitle*(title) | Set the title of the panel. |

### 3.5.2 Setup Panel Option Button

| | |
|---|---|
| *RibbonPanel.panelOptionButton*() | Return the panel option button. |
| *RibbonPanel.setPanelOptionToolTip*(text) | Set the tooltip of the panel option button. |
| *RibbonPanel.panelOptionClicked*(bool) | int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL |

### 3.5.3 Add Widgets to Panels

| | |
|---|---|
| *RibbonPanel.addWidget*(widget, *[, rowSpan, ...]) | Add a widget to the panel. |
| *RibbonPanel.addWidgetsBy*(data) | Add widgets to the panel. |
| *RibbonPanel.removeWidget*(widget) | Remove a widget from the panel. |
| *RibbonPanel.widget*(index) | Get the widget at the given index. |
| *RibbonPanel.widgets*() | Get all the widgets in the panel. |
| *RibbonPanel.addSmallWidget*(widget, *[, ...]) | |
| *RibbonPanel.addMediumWidget*(widget, *[, ...]) | |
| *RibbonPanel.addLargeWidget*(widget, *[, ...]) | |
| *RibbonPanel.addButton*([text, icon, ...]) | Add a button to the panel. |
| *RibbonPanel.addSmallButton*([text, icon, ...]) | |
| *RibbonPanel.addMediumButton*([text, icon, ...]) | |
| *RibbonPanel.addLargeButton*([text, icon, ...]) | |
| *RibbonPanel.addToggleButton*([text, icon, ...]) | |

continues on next page

Table 1 – continued from previous page

| | |
|---|---|
| RibbonPanel.addSmallToggleButton([text, ...]) | |
| RibbonPanel.addMediumToggleButton([text, ...]) | |
| RibbonPanel.addLargeToggleButton([text, ...]) | |
| RibbonPanel.addComboBox(*args, **kwargs) | |
| RibbonPanel.addFontComboBox(*args, **kwargs) | |
| RibbonPanel.addLineEdit(*args, **kwargs) | |
| RibbonPanel.addTextEdit(*args, **kwargs) | |
| RibbonPanel.addPlainTextEdit(*args, **kwargs) | |
| RibbonPanel.addLabel(*args, **kwargs) | |
| RibbonPanel.addProgressBar(*args, **kwargs) | |
| RibbonPanel.addSlider(*args, **kwargs) | |
| RibbonPanel.addSpinBox(*args, **kwargs) | |
| RibbonPanel.addDoubleSpinBox(*args, **kwargs) | |
| RibbonPanel.addDateEdit(*args, **kwargs) | |
| RibbonPanel.addTimeEdit(*args, **kwargs) | |
| RibbonPanel.addDateTimeEdit(*args, **kwargs) | |
| RibbonPanel.addTableWidget(*args, **kwargs) | |
| RibbonPanel.addTreeWidget(*args, **kwargs) | |
| RibbonPanel.addListWidget(*args, **kwargs) | |
| RibbonPanel.addCalendarWidget(*args, **kwargs) | |
| RibbonPanel.addSeparator([orientation, width]) | Add a separator to the panel. |
| RibbonPanel.addHorizontalSeparator(*[, ...]) | |
| RibbonPanel.addVerticalSeparator(*[, ...]) | |
| RibbonPanel.addGallery([minimumWidth, ...]) | Add a gallery to the panel. |

### 3.5.4 Example

For example, using the following code,

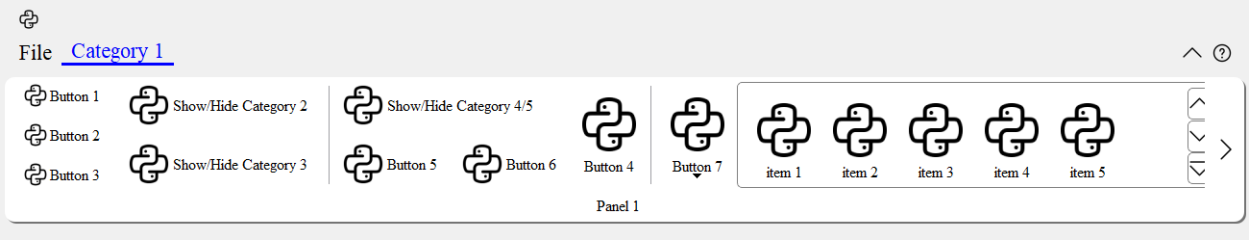
```
import sys

from qtpy import QtGui
from qtpy.QtCore import Qt
from qtpy.QtGui import QIcon
from qtpy.QtWidgets import QApplication, QLabel, QLineEdit, QMenu, QToolButton

from pyqtribbon import RibbonBar
from pyqtribbon.screenshotwindow import RibbonScreenShotWindow

if __name__ == "__main__":
    app = QApplication(sys.argv)
    app.setFont(QtGui.QFont("Times New Roman", 8))
    window = RibbonScreenShotWindow("panel.png")

    # Ribbon bar
    ribbonbar = RibbonBar()
    window.setMenuBar(ribbonbar)

    category1 = ribbonbar.addCategory("Category 1")
    panel = category1.addPanel("Panel 1", showPanelOptionButton=False)
    panel.addSmallButton("Button 1", icon=QIcon("python.png"))
    panel.addSmallButton("Button 2", icon=QIcon("python.png"))
    panel.addSmallButton("Button 3", icon=QIcon("python.png"))
    panel.addMediumToggleButton("Show/Hide Category 2", icon=QIcon("python.png"))
    panel.addVerticalSeparator()
    panel.addMediumToggleButton("Show/Hide Category 3", icon=QIcon("python.png"))
    panel.addMediumToggleButton("Show/Hide Category 4/5", icon=QIcon("python.png"),␣
↪colSpan=2, alignment=Qt.AlignLeft)
    panel.addLargeButton("Button 4", icon=QIcon("python.png"))
    panel.addVerticalSeparator()
    panel.addMediumButton("Button 5", icon=QIcon("python.png"))
    panel.addMediumButton("Button 6", icon=QIcon("python.png"))

    button = panel.addLargeButton("Button 7", icon=QIcon("python.png"))
    menu = QMenu()
    menu.addAction(QIcon("python.png"), "Action 1")
    menu.addAction(QIcon("python.png"), "Action 2")
    menu.addAction(QIcon("python.png"), "Action 3")
    button.setMenu(menu)
    button.setPopupMode(QToolButton.InstantPopup)
    panel.addWidget(button, rowSpan=6)

    gallery = panel.addGallery(minimumWidth=500, popupHideOnClick=True)
    for i in range(100):
        gallery.addToggleButton(f"item {i+1}", QIcon("python.png"))
    popupMenu = gallery.popupMenu()
    submenu = popupMenu.addMenu(QIcon("python.png"), "Submenu")
    submenu.addAction(QIcon("python.png"), "Action 4")
```

(continues on next page)

(continued from previous page)

```
    popupMenu.addAction(QIcon("python.png"), "Action 1")
    popupMenu.addAction(QIcon("python.png"), "Action 2")
    popupMenu.addSeparator()
    popupMenu.addWidget(QLabel("This is a custom widget"))
    formLayout = popupMenu.addFormLayoutWidget()
    formLayout.addRow(QLabel("Row 1"), QLineEdit())

    # Show the window
    window.resize(1300, 250)
    window.show()

    sys.exit(app.exec_())
```

You can get a window like this:




## 3.6 A Complete Example

The following code snippet is a complete example.

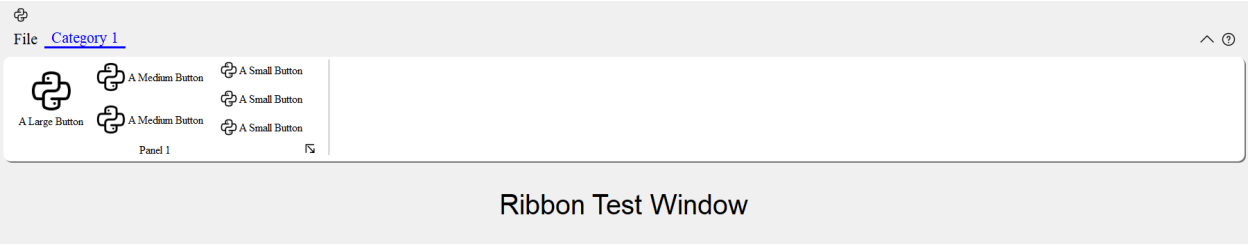
```
import sys

from PyQt5.QtCore import Qt
from PyQt5.QtGui import QFont, QIcon
from PyQt5.QtWidgets import QApplication, QLabel, QVBoxLayout, QWidget

from pyqtribbon import RibbonBar
from pyqtribbon.screenshotwindow import RibbonScreenShotWindow
from pyqtribbon.utils import DataFile

if __name__ == "__main__":
    app = QApplication(sys.argv)
    app.setFont(QFont("Times New Roman", 8))

    # Central widget
    window = RibbonScreenShotWindow("tutorial-ribbonbar.png")
    window.setWindowIcon(QIcon(DataFile("icons/python.png")))
    centralWidget = QWidget()
    window.setCentralWidget(centralWidget)
    layout = QVBoxLayout(centralWidget)

    # Ribbon bar
    ribbonbar = RibbonBar()
```

(continues on next page)

(continued from previous page)

```
    window.setMenuBar(ribbonbar)
    category = ribbonbar.addCategory("Category 1")
    panel = category.addPanel("Panel 1")
    panel.addLargeButton("A Large Button", QIcon(DataFile("icons/python.png")))
    panel.addMediumButton("A Medium Button", QIcon(DataFile("icons/python.png")))
    panel.addMediumButton("A Medium Button", QIcon(DataFile("icons/python.png")))
    panel.addSmallButton("A Small Button", QIcon(DataFile("icons/python.png")))
    panel.addSmallButton("A Small Button", QIcon(DataFile("icons/python.png")))
    panel.addSmallButton("A Small Button", QIcon(DataFile("icons/python.png")))

    # Display a label in the main window
    label = QLabel("Ribbon Test Window")
    label.setFont(QFont("Arial", 20))
    label.setAlignment(Qt.AlignCenter)

    # Add the ribbon bar and label to the layout
    layout.addWidget(label, 1)

    # Show the window
    window.resize(1800, 350)  # type: ignore
    window.show()
    sys.exit(app.exec_())
```

You can get a window like this:



CHAPTER

FOUR

# API REFERENCES

## 4.1 pyqtribbon package

### 4.1.1 Submodules

#### pyqtribbon.category module

class pyqtribbon.category.**RibbonCategory**(*title: str = ''*, *style:* RibbonCategoryStyle = *RibbonCategoryStyle.Normal*, *color: QColor = None*, *parent=None*)

class pyqtribbon.category.**RibbonCategory**(*parent=None*)

Bases: *RibbonCategoryLayoutWidget*

The RibbonCategory is the logical grouping that represents the contents of a ribbon tab.

Create a new category.

**Parameters**

- **title** – The title of the category.
- **style** – The button style of the category.
- **color** – The color of the context category.
- **parent** – The parent widget.

**addPanel**(*title: str*, *showPanelOptionButton=True*) → *RibbonPanel*

Add a new panel to the category.

**Parameters**

- **title** – The title of the panel.
- **showPanelOptionButton** – Whether to show the panel option button.

**Returns**

The newly created panel.

**addPanelsBy**(*data: Dict[str, Dict]*) → Dict[str, *RibbonPanel*]

Add panels from a dictionary.

**Parameters**

**data** – The dictionary. The keys are the titles of the panels. The value is a dictionary of arguments. the argument showPanelOptionButton is a boolean to decide whether to show the panel option button, the rest arguments are passed to the RibbonPanel.addWidgetsBy() method. The dict is of the form:

```
{
    "panel-title": {
        "showPanelOptionButton": True,
        "widgets": {
            "widget-name": {
                "type": "Button",
                "arguments": {
                    "key1": "value1",
                    "key2": "value2"
                }
            },
        }
    },
}
```

**Returns**
A dictionary of the newly created panels.

**categoryStyle**() → *RibbonCategoryStyle*

Return the button style of the category.

**Returns**
The button style.

**panel**(*title: str*) → *RibbonPanel*

Return a panel from the category.

**Parameters**
**title** – The title of the panel.

**Returns**
The panel.

**panels**() → Dict[str, *RibbonPanel*]

Return all panels in the category.

**Returns**
The panels.

**removePanel**(*title: str*)

Remove a panel from the category.

**Parameters**
**title** – The title of the panel.

**setCategoryStyle**(*style:* RibbonCategoryStyle)

Set the button style of the category.

**Parameters**
**style** – The button style.

**setMaximumRows**(*rows: int*)

Set the maximum number of rows.

**Parameters**
**rows** – The maximum number of rows.

**takePanel**(*title: str*) → *RibbonPanel*

Remove and return a panel from the category.

**Parameters**
**title** – The title of the panel.

**Returns**
The removed panel.

**title**() → str

Return the title of the category.

**class** pyqtribbon.category.**RibbonCategoryLayoutButton**

Bases: QToolButton

Previous/Next buttons in the category when the size is not enough for the widgets.

**class** pyqtribbon.category.**RibbonCategoryLayoutWidget**(*parent=None*)

Bases: QFrame

The category layout widget's category scroll area to arrange the widgets in the category.

Create a new category layout widget.

**Parameters**
**parent** – The parent widget.

**addWidget**(*widget: QWidget*)

Add a widget to the category layout.

**Parameters**
**widget** – The widget to add.

**autoSetScrollButtonsVisible**()

Set the visibility of the scroll buttons.

**displayOptionsButtonClicked**

int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

types is normally a sequence of individual types. Each type is either a type object or a string that is the name of a C++ type. Alternatively each type could itself be a sequence of types each describing a different overloaded signal. name is the optional C++ name of the signal. If it is not specified then the name of the class attribute that is bound to the signal is used. revision is the optional revision of the signal that is exported to QML. If it is not specified then 0 is used. arguments is the optional sequence of the names of the signal's arguments.

**Type**
pyqtSignal(*types, name

**Type**
str = ..., revision

**paintEvent**(*a0: QPaintEvent*) → None

Override the paint event to draw the background.

**removeWidget**(*widget: QWidget*)

Remove a widget from the category layout.

**Parameters**
**widget** – The widget to remove.

**resizeEvent**(*a0: QResizeEvent*) → None

Override the resize event to resize the scroll area.

**scrollNext**()

Scroll the category to the next widget.

**scrollPrevious**()

Scroll the category to the previous widget.

**takeWidget**(*widget: QWidget*) → QWidget

Remove and return a widget from the category layout.

**Parameters**
**widget** – The widget to remove.

**Returns**
The widget that was removed.

**class** pyqtribbon.category.**RibbonCategoryScrollArea**

Bases: QScrollArea

Scroll area for the gallery

**class** pyqtribbon.category.**RibbonCategoryScrollAreaContents**

Bases: QFrame

Scroll area contents for the gallery

**class** pyqtribbon.category.**RibbonContextCategories**(*name: str*, *color: QColor*, *categories: Dict[str, RibbonContextCategory]*, *ribbon*)

Bases: Dict[str, *RibbonContextCategory*]

A list of context categories.

**categoriesVisible**() → bool

Return whether the categories are shown.

**color**() → QColor

Return the color of the context categories.

**hideContextCategories**()

Hide the categories

**name**() → str

Return the name of the context categories.

**setCategoriesVisible**(*visible: bool*)

Set the state of the categories.

**setColor**(*color: QColor*)

Set the color of the context categories.

**setName**(*name: str*)

Set the name of the context categories.

**showContextCategories**()

Show the categories

class pyqtribbon.category.**RibbonContextCategory**(*title: str*, *color: QColor*, *parent: QWidget*)

Bases: *RibbonCategory*

A context category.

Create a new context category.

**Parameters**

- **title** – The title of the category.
- **color** – The color of the context category.
- **parent** – The parent widget.

**categoryVisible**() → bool

Return whether the category is shown.

**Returns**
Whether the category is shown.

**color**() → QColor

Return the color of the context category.

**Returns**
The color of the context category.

**hideContextCategory**()

Hide the given category, if it is not a context category, nothing happens.

**setCategoryStyle**(*style:* RibbonCategoryStyle)

Set the button style of the category.

**Parameters**
**style** – The button style.

**setCategoryVisible**(*visible: bool*)

Set the state of the category.

**Parameters**
**visible** – The state.

**setColor**(*color: QColor*)

Set the color of the context category.

**Parameters**
**color** – The color of the context category.

**showContextCategory**()

Show the given category, if it is not a context category, nothing happens.

class pyqtribbon.category.**RibbonNormalCategory**(*title: str*, *parent: QWidget*)

Bases: *RibbonCategory*

A normal category.

Create a new normal category.

**Parameters**

- **title** – The title of the category.
- **parent** – The parent widget.

**setCategoryStyle**(*style:* RibbonCategoryStyle)

Set the button style of the category.

**Parameters**
**style** – The button style.

### pyqtribbon.constants module

**class** pyqtribbon.constants.**RibbonButtonStyle**(*value*)

Bases: IntEnum

Button style, Small, Medium, or Large.

**Large = 2**

**Medium = 1**

**Small = 0**

**class** pyqtribbon.constants.**RibbonCategoryStyle**(*value*)

Bases: IntEnum

The button style of a category.

**Context = 1**

**Normal = 0**

**class** pyqtribbon.constants.**RibbonSpaceFindMode**(*value*)

Bases: IntEnum

Mode to find available space in a grid layout, ColumnWise or RowWise.

**ColumnWise = 0**

**RowWise = 1**

**class** pyqtribbon.constants.**RibbonStyle**(*value*)

Bases: IntEnum

An enumeration.

**Debug = 1**

**Default = 0**

pyqtribbon.constants.**contextColors = [<PyQt5.QtGui.QColor object>, <PyQt5.QtGui.QColor object>, <PyQt5.QtGui.QColor object>, <PyQt5.QtGui.QColor object>, <PyQt5.QtGui.QColor object>, <PyQt5.QtGui.QColor object>]**

A list of context category colors

### pyqtribbon.gallery module

**class** pyqtribbon.gallery.**RibbonGallery**(*minimumWidth=800*, *popupHideOnClick=False*, *parent=None*)

**class** pyqtribbon.gallery.**RibbonGallery**(*parent=None*)

Bases: QFrame

A widget that displays a gallery of buttons.

Create a gallery.

**Parameters**

- **minimumWidth** – minimum width of the gallery
- **popupHideOnClick** – hide on click flag
- **parent** – parent widget

**addButton**(*text: str | None = None*, *icon: QIcon | None = None*, *slot=None*, *shortcut=None*, *tooltip=None*, *statusTip=None*, *checkable=False*) → RibbonToolButton

Add a button to the gallery

**Parameters**

- **text** – text of the button
- **icon** – icon of the button
- **slot** – slot to call when the button is clicked
- **shortcut** – shortcut of the button
- **tooltip** – tooltip of the button
- **statusTip** – status tip of the button
- **checkable** – checkable flag of the button.

**Returns**
the button added

**addToggleButton**(*text: str | None = None*, *icon: QIcon | None = None*, *slot=None*, *shortcut=None*, *tooltip=None*, *statusTip=None*) → RibbonToolButton

Add a toggle button to the gallery

**Parameters**

- **text** – text of the button
- **icon** – icon of the button
- **slot** – slot to call when the button is clicked
- **shortcut** – shortcut of the button
- **tooltip** – tooltip of the button
- **statusTip** – status tip of the button.

**Returns**
the button added

**hidePopupWidget**()

Hide the popup window

**popupMenu**() → *RibbonPermanentMenu*

Return the popup menu.

**popupWindowSize**()

Return the size of the popup window

**Returns**
size of the popup window

**resizeEvent**(*a0: QResizeEvent*) → None

Resize the gallery.

**setPopupHideOnClick**(*popupHideOnClick: bool*)

Set the hide on click flag

**Parameters**
**popupHideOnClick** – hide on click flag

**setPopupWindowSize**(*size: QSize*)

Set the size of the popup window

**Parameters**
**size** – size of the popup window

**setSelectedButton**()

Set the selected button

**showPopup**()

Show the popup window

**class** pyqtribbon.gallery.**RibbonGalleryButton**

Bases: QToolButton

Gallery button.

**class** pyqtribbon.gallery.**RibbonGalleryListWidget**(*parent=None*)

Bases: QListWidget

Gallery list widget.

**resizeEvent**(*e: QResizeEvent*) → None

Resize the list widget.

**scrollToNextRow**() → None

Scroll to the next row.

**scrollToPreviousRow**() → None

Scroll to the previous row.

**class** pyqtribbon.gallery.**RibbonGalleryPopupListWidget**(*parent=None*)

Bases: *RibbonGalleryListWidget*

Gallery popup list widget.

**class** pyqtribbon.gallery.**RibbonPopupWidget**

Bases: QFrame

The popup widget for the gallery widget.

### pyqtribbon.logger module

https://timlehr.com/python-exception-hooks-with-qt-message-box/

class pyqtribbon.logger.**UncaughtHook**(**args*, ***kwargs*)

Bases: QObject

**exception_hook**(*exc_type*, *exc_value*, *exc_traceback*)

Function handling uncaught exceptions. It is triggered each time an uncaught exception occurs.

static **show_exception_box**(*log_msg*)

Checks if a QApplication instance is available and shows a messagebox with the exception message. If unavailable (non-console application), log an additional notice.

### pyqtribbon.menu module

class pyqtribbon.menu.**RibbonMenu**(*title: str = ''*, *parent=None*)

class pyqtribbon.menu.**RibbonMenu**(*parent=None*)

Bases: QMenu

Create a new panel.

**Parameters**

- **title** – The title of the menu.
- **parent** – The parent widget.

**addFormLayoutWidget**() → QFormLayout

Add a form layout widget to the menu.

**Returns**
The form layout.

**addGridLayoutWidget**() → QGridLayout

Add a grid layout widget to the menu.

**Returns**
The grid layout.

**addHorizontalLayoutWidget**() → QHBoxLayout

Add a horizontal layout widget to the menu.

**Returns**
The horizontal layout.

**addLabel**(*text: str = ''*, *alignment=1*)

Add a label to the menu.

**Parameters**

- **text** – The text of the label.
- **alignment** – The alignment of the label.

**addSpacing**(*spacing: int = 5*)

Add spacing to the menu.

**Parameters**
**spacing** – The spacing.

**addVerticalLayoutWidget**() → QVBoxLayout

Add a vertical layout widget to the menu.

**Returns**
The vertical layout.

**addWidget**(*widget: QWidget*)

Add a widget to the menu.

**Parameters**
**widget** – The widget to add.

class pyqtribbon.menu.**RibbonPermanentMenu**(*title: str = ''*, *parent=None*)

class pyqtribbon.menu.**RibbonPermanentMenu**(*parent=None*)

Bases: *RibbonMenu*

A permanent menu.

Create a new panel.

**Parameters**

- **title** – The title of the menu.
- **parent** – The parent widget.

**actionAdded**(*QtWidgets.QAction*)

int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

types is normally a sequence of individual types. Each type is either a type object or a string that is the name of a C++ type. Alternatively each type could itself be a sequence of types each describing a different overloaded signal. name is the optional C++ name of the signal. If it is not specified then the name of the class attribute that is bound to the signal is used. revision is the optional revision of the signal that is exported to QML. If it is not specified then 0 is used. arguments is the optional sequence of the names of the signal's arguments.

**Type**
pyqtSignal(*types, name

**Type**
str = ..., revision

**addAction**(*self*, *action: QAction | None*)

**hideEvent**(*self*, *a0: QHideEvent | None*)

## pyqtribbon.panel module

class pyqtribbon.panel.**RibbonGridLayoutManager**(*rows: int*)

Bases: object

Grid Layout Manager.

Create a new grid layout manager.

**Parameters**
**rows** – The number of rows in the grid layout.

**request_cells**(*rowSpan: int = 1*, *colSpan: int = 1*, *mode=RibbonSpaceFindMode.ColumnWise*)

Request a number of available cells from the grid.

**Parameters**

- **rowSpan** – The number of rows the cell should span.
- **colSpan** – The number of columns the cell should span.
- **mode** – The mode of the grid.

**Returns**

row, col, the row and column of the requested cell.

class pyqtribbon.panel.**RibbonPanel**(*title: str = ''*, *maxRows: int = 6*, *showPanelOptionButton=True*, *parent=None*)

class pyqtribbon.panel.**RibbonPanel**(*parent=None*)

Bases: QFrame

Panel in the ribbon category.

Create a new panel.

**Parameters**

- **title** – The title of the panel.
- **maxRows** – The maximal number of rows in the panel.
- **showPanelOptionButton** – Whether to show the panel option button.
- **parent** – The parent widget.

**addButton**(*text: str | None = None*, *icon: QIcon | None = None*, *showText: bool = True*, *slot: Callable | None = None*, *shortcut: QKeySequence | None = None*, *tooltip: str | None = None*, *statusTip: str | None = None*, *checkable: bool = False*, ***, *rowSpan:* RibbonButtonStyle *= RibbonButtonStyle.Large*, ***kwargs*) → *RibbonToolButton*

Add a button to the panel.

**Parameters**

- **text** – The text of the button.
- **icon** – The icon of the button.
- **showText** – Whether to show the text of the button.
- **slot** – The slot to call when the button is clicked.
- **shortcut** – The shortcut of the button.
- **tooltip** – The tooltip of the button.
- **statusTip** – The status tip of the button.
- **checkable** – Whether the button is checkable.
- **rowSpan** – The type of the button corresponding to the number of rows it should span.
- **kwargs** – keyword arguments to control the properties of the widget on the ribbon bar.

**Returns**

The button that was added.

**addCalendarWidget**(**args*, ***kwargs*) → QWidget

**addComboBox**(**args*, ***kwargs*) → QWidget

**addDateEdit**(**args*, ***kwargs*) → QWidget

**addDateTimeEdit**(**args*, ***kwargs*) → QWidget

**addDoubleSpinBox**(**args*, ***kwargs*) → QWidget

**addFontComboBox**(**args*, ***kwargs*) → QWidget

**addGallery**(*minimumWidth=800*, *popupHideOnClick=False*, ***kwargs*) → *RibbonGallery*

Add a gallery to the panel.

**Parameters**

- **minimumWidth** – The minimum width of the gallery.
- **popupHideOnClick** – Whether the gallery popup should be hidden when a user clicks on it.
- **kwargs** – keyword arguments to control the properties of the widget on the ribbon bar.

**Returns**

The gallery.

**addHorizontalSeparator**(***, *orientation=1*, *width=6*, ***kwargs*) → *RibbonSeparator*

**addLabel**(**args*, ***kwargs*) → QWidget

**addLargeButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, *checkable: bool = False*, ***, *rowSpan:* RibbonButtonStyle *= RibbonButtonStyle.Large*, ***kwargs*) → *RibbonToolButton*

**addLargeToggleButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, ***, *checkable: bool = True*, *rowSpan:* RibbonButtonStyle *= RibbonButtonStyle.Large*, ***kwargs*) → *RibbonToolButton*

**addLargeWidget**(*widget: QWidget*, ***, *rowSpan: int* | RibbonButtonStyle *= RibbonButtonStyle.Large*, *colSpan: int = 1*, *mode=RibbonSpaceFindMode.ColumnWise*, *alignment=132*, *fixedHeight: bool* | *float = False*) → QWidget | Any

**addLineEdit**(**args*, ***kwargs*) → QWidget

**addListWidget**(**args*, ***kwargs*) → QWidget

**addMediumButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, *checkable: bool = False*, ***, *rowSpan:* RibbonButtonStyle *= RibbonButtonStyle.Medium*, ***kwargs*) → *RibbonToolButton*

**addMediumToggleButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, ***, *checkable: bool = True*, *rowSpan:* RibbonButtonStyle *= RibbonButtonStyle.Medium*, ***kwargs*) → *RibbonToolButton*

**addMediumWidget**(*widget: QWidget*, *, *rowSpan: int | RibbonButtonStyle = RibbonButtonStyle.Medium*, *colSpan: int = 1*, *mode=RibbonSpaceFindMode.ColumnWise*, *alignment=132*, *fixedHeight: bool | float = False*) → QWidget | Any

**addPlainTextEdit**(**args*, ***kwargs*) → QWidget

**addProgressBar**(**args*, ***kwargs*) → QWidget

**addSeparator**(*orientation=2*, *width=6*, ***kwargs*) → *RibbonSeparator*

Add a separator to the panel.

**Parameters**

- **orientation** – The orientation of the separator.
- **width** – The width of the separator.
- **kwargs** – keyword arguments to control the properties of the widget on the ribbon bar.

**Returns**

The separator.

**addSlider**(**args*, ***kwargs*) → QWidget

**addSmallButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, *checkable: bool = False*, *, *rowSpan: RibbonButtonStyle = RibbonButtonStyle.Small*, ***kwargs*) → *RibbonToolButton*

**addSmallToggleButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, *, *checkable: bool = True*, *rowSpan: RibbonButtonStyle = RibbonButtonStyle.Small*, ***kwargs*) → *RibbonToolButton*

**addSmallWidget**(*widget: QWidget*, *, *rowSpan: int | RibbonButtonStyle = RibbonButtonStyle.Small*, *colSpan: int = 1*, *mode=RibbonSpaceFindMode.ColumnWise*, *alignment=132*, *fixedHeight: bool | float = False*) → QWidget | Any

**addSpinBox**(**args*, ***kwargs*) → QWidget

**addTableWidget**(**args*, ***kwargs*) → QWidget

**addTextEdit**(**args*, ***kwargs*) → QWidget

**addTimeEdit**(**args*, ***kwargs*) → QWidget

**addToggleButton**(*text: str = None*, *icon: QIcon = None*, *showText: bool = True*, *slot: Callable = None*, *shortcut: QKeySequence = None*, *tooltip: str = None*, *statusTip: str = None*, *, *checkable: bool = True*, *rowSpan: RibbonButtonStyle = RibbonButtonStyle.Large*, ***kwargs*) → *RibbonToolButton*

**addTreeWidget**(**args*, ***kwargs*) → QWidget

**addVerticalSeparator**(*, *orientation=2*, *width=6*, ***kwargs*) → *RibbonSeparator*

**addWidget**(*widget: QWidget*, *, *rowSpan: int | RibbonButtonStyle = RibbonButtonStyle.Small*, *colSpan: int = 1*, *mode=RibbonSpaceFindMode.ColumnWise*, *alignment=132*, *fixedHeight: bool | float = False*) → QWidget | Any

Add a widget to the panel.

**Parameters**

- **widget** – The widget to add.
- **rowSpan** – The number of rows the widget should span, 2: small, 3: medium, 6: large.
- **colSpan** – The number of columns the widget should span.
- **mode** – The mode to find spaces.
- **alignment** – The alignment of the widget.
- **fixedHeight** – Whether to fix the height of the widget, it can be a boolean, a percentage or a fixed height, when a boolean is given, the height is fixed to the maximum height allowed if the value is True, when a percentage is given (0 < percentage < 1) the height is calculated from the height of the maximum height allowed, depends on the number of rows to span. The minimum height is 40% of the maximum height allowed.

**Returns**

The added widget.

**addWidgetsBy**(*data: Dict[str, Dict]*) → Dict[str, QWidget]

Add widgets to the panel.

**Parameters**

**data** – The data to add. The dict is of the form:

```
{
    "widget-name": {
        "type": "Button",
        "arguments": {
            "key1": "value1",
            "key2": "value2"
        }
    },
}
```

Possible types are: Button, SmallButton, MediumButton, LargeButton, ToggleButton, SmallToggleButton, MediumToggleButton, LargeToggleButton, ComboBox, FontComboBox, LineEdit, TextEdit, PlainTextEdit, Label, ProgressBar, SpinBox, DoubleSpinBox, DataEdit, TimeEdit, DateTimeEdit, TableWidget, TreeWidget, ListWidget, CalendarWidget, Separator, HorizontalSeparator, VerticalSeparator, Gallery.

**Returns**

A dictionary of the added widgets.

**defaultRowSpan**(*rowSpan: int | RibbonButtonStyle*) → int

Return the number of span rows for the given widget type.

**Parameters**

**rowSpan** – row span or type.

**Returns**

The number of span rows for the given widget type.

**largeRows**() → int

Return the number of span rows for large widgets.

**Returns**

The number of span rows for large widgets.

**maximumRows**() → int

Return the maximal number of rows in the panel.

**Returns**
The maximal number of rows in the panel.

**mediumRows**() → int

Return the number of span rows for medium widgets.

**Returns**
The number of span rows for medium widgets.

**panelOptionButton**() → *RibbonPanelOptionButton*

Return the panel option button.

**Returns**
The panel option button.

**panelOptionClicked**(*bool*)

int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

types is normally a sequence of individual types. Each type is either a type object or a string that is the name of a C++ type. Alternatively each type could itself be a sequence of types each describing a different overloaded signal. name is the optional C++ name of the signal. If it is not specified then the name of the class attribute that is bound to the signal is used. revision is the optional revision of the signal that is exported to QML. If it is not specified then 0 is used. arguments is the optional sequence of the names of the signal's arguments.

**Type**
pyqtSignal(*types, name

**Type**
str = ..., revision

**removeWidget**(*widget: QWidget*)

Remove a widget from the panel.

**ribbonArguments = ['rowSpan', 'colSpan', 'mode', 'alignment', 'fixedHeight']**

**rowHeight**() → int

Return the height of a row.

**setLargeRows**(*rows: int*)

Set the number of span rows for large widgets.

**Parameters**
**rows** – The number of span rows for large widgets.

**setMaximumRows**(*maxRows: int*)

Set the maximal number of rows in the panel.

**Parameters**
**maxRows** – The maximal number of rows in the panel.

**setMediumRows**(*rows: int*)

Set the number of span rows for medium widgets.

**Parameters**
**rows** – The number of span rows for medium widgets.

**setPanelOptionToolTip**(*text: str*)

Set the tooltip of the panel option button.

**Parameters**
**text** – The tooltip text.

**setSmallRows**(*rows: int*)

Set the number of span rows for small widgets.

**Parameters**
**rows** – The number of span rows for small widgets.

**setTitle**(*title: str*)

Set the title of the panel.

**Parameters**
**title** – The title to set.

**smallRows**() → int

Return the number of span rows for small widgets.

**Returns**
The number of span rows for small widgets.

**title**()

Get the title of the panel.

**Returns**
The title.

**widget**(*index: int*) → QWidget

Get the widget at the given index.

**Parameters**
**index** – The index of the widget, starting from 0.

**Returns**
The widget at the given index.

**widgets**() → List[QWidget]

Get all the widgets in the panel.

**Returns**
A list of all the widgets in the panel.

class pyqtribbon.panel.**RibbonPanelItemWidget**(*parent=None*)

Bases: QFrame

Widget to display a panel item.

Create a new panel item.

**Parameters**
**parent** – The parent widget.

**addWidget**(*widget*)

Add a widget to the panel item.

**Parameters**
**widget** – The widget to add.

**class** pyqtribbon.panel.**RibbonPanelOptionButton**

Bases: QToolButton

Button to display the options of a panel.

**class** pyqtribbon.panel.**RibbonPanelTitle**

Bases: QLabel

Widget to display the title of a panel.

### pyqtribbon.ribbonbar module

**class** pyqtribbon.ribbonbar.**RibbonBar**(*title: str = 'Ribbon Bar Title'*, *maxRows=6*, *parent=None*)
**class** pyqtribbon.ribbonbar.**RibbonBar**(*parent=None*)

Bases: QMenuBar

The RibbonBar class is the top level widget that contains the ribbon.

Create a new ribbon.

**Parameters**

- **title** – The title of the ribbon.
- **maxRows** – The maximum number of rows.
- **parent** – The parent widget of the ribbon.

**actionAt**(*self*, *a0: QPoint*) → QAction | None

**actionGeometry**(*self*, *a0: QAction | None*) → QRect

**activeAction**(*self*) → QAction | None

**addAction**(*self*, *action: QAction | None*)
**addAction**(*self*, *text: str | None*) → QAction | None
**addAction**(*self*, *text: str | None*, *slot: PYQT_SLOT*) → QAction | None

**addCategoriesBy**(*data: Dict[str, Dict]*) → Dict[str, *RibbonCategory*]

Add categories from a dict.

**Parameters**

**data** – The dict of categories. The dict is of the form:

```
{
    "category-title": {
        "style": RibbonCategoryStyle.Normal,
        "color": QtCore.Qt.red,
        "panels": {
            "panel-title": {
                "showPanelOptionButton": True,
                "widgets": {
                    "widget-name": {
                        "type": "Button",
                        "arguments": {
                            "key1": "value1",
                            "key2": "value2"
                        }
```

(continues on next page)

(continued from previous page)

```
                    },
                }
            },
        },
    }
}
```

**Returns**
A dict of categories of the ribbon.

**addCategory**(*title: str*, *style=RibbonCategoryStyle.Normal*, *color: QColor | None = None*) → *RibbonNormalCategory | RibbonContextCategory*

Add a new category to the ribbon.

**Parameters**

- **title** – The title of the category.
- **style** – The button style of the category.
- **color** – The color of the context category, only used if style is Context, if None, the default color will be used.

**Returns**
The newly created category.

**addContextCategories**(*name: str*, *titles: List[str]*, *color: QColor | GlobalColor = 9*) → *RibbonContextCategories*

Add a group of context categories with the same tab color to the ribbon.

**Parameters**

- **name** – The name of the context categories.
- **titles** – The title of the category.
- **color** – The color of the context category, if None, the default color will be used.

**Returns**
The newly created category.

**addContextCategory**(*title: str*, *color: QColor | GlobalColor = 9*) → *RibbonContextCategory*

Add a new context category to the ribbon.

**Parameters**

- **title** – The title of the category.
- **color** – The color of the context category, if None, the default color will be used.

**Returns**
The newly created category.

**addFileMenu**() → *RibbonMenu*

Add a file menu to the ribbon.

**addMenu**(*self*, *menu: QMenu | None*) → QAction | None

**addMenu**(*self*, *title: str | None*) → QMenu | None

**addMenu**(*self*, *icon: QIcon*, *title: str | None*) → QMenu | None

**addNormalCategory**(*title: str*) → *RibbonNormalCategory*

Add a new category to the ribbon.

**Parameters**
**title** – The title of the category.

**Returns**
The newly created category.

**addQuickAccessButton**(*button: QToolButton*)

Add a button to the quick access bar.

**Parameters**
**button** – The button to add.

**addRightToolButton**(*button: QToolButton*)

Add a widget to the right button bar.

**Parameters**
**button** – The button to add.

**addSeparator**(*self*) → QAction | None

**addTitleWidget**(*widget: QWidget*)

Add a widget to the title widget.

**Parameters**
**widget** – The widget to add.

**applicationOptionButton**() → *RibbonApplicationButton*

Return the application button.

**categories**() → Dict[str, *RibbonCategory*]

Return a list of categories of the ribbon.

**Returns**
A dict of categories of the ribbon.

**category**(*name: str*) → *RibbonCategory*

Return the category with the given name.

**Parameters**
**name** – The name of the category.

**Returns**
The category with the given name.

**categoryVisible**(*category:* RibbonCategory) → bool

Return whether the category is shown.

**Parameters**
**category** – The category to check.

**Returns**
Whether the category is shown.

**clear**(*self*)

**collapseRibbonButton**() → QToolButton

Return the collapse ribbon button.

**Returns**
The collapse ribbon button.

**cornerWidget**(*self*, *corner: Corner = Qt.TopRightCorner*) → QWidget | None

**currentCategory**() → *RibbonCategory*

Return the current category.

**Returns**
The current category.

**eventFilter**(*self*, *a0: QObject | None*, *a1: QEvent | None*) → bool

**helpButtonClicked**(*bool*)

Signal, the help button was clicked.

**helpRibbonButton**() → QToolButton

Return the help button of the ribbon.

**Returns**
The help button of the ribbon.

**hideContextCategory**(*category:* RibbonContextCategory | RibbonContextCategories)

Hide the given category or categories, if it is not a context category, nothing happens.

**Parameters**
**category** – The category to hide.

**hideRibbon**()

Hide the ribbon.

**insertMenu**(*self*, *before: QAction | None*, *menu: QMenu | None*) → QAction | None

**insertSeparator**(*self*, *before: QAction | None*) → QAction | None

**insertTitleWidget**(*index: int*, *widget: QWidget*)

Insert a widget to the title widget.

**Parameters**

- **index** – The index to insert the widget.
- **widget** – The widget to insert.

**isDefaultUp**(*self*) → bool

**isNativeMenuBar**(*self*) → bool

**minimumSizeHint**() → QSize

Return the minimum size hint of the widget.

**Returns**
The minimum size hint.

**quickAccessToolBar**() → QToolBar

Return the quick access toolbar of the ribbon.

**Returns**
The quick access toolbar of the ribbon.

**removeCategories**(*categories:* RibbonContextCategories)

Remove a list of categories from the ribbon.

**Parameters**
**categories** – The categories to remove.

**removeCategory**(*category:* RibbonCategory)

Remove a category from the ribbon.

**Parameters**
**category** – The category to remove.

**removeCollapseButton**()

Remove the min button from the ribbon.

**removeHelpButton**()

Remove the help button from the ribbon.

**removeTitleWidget**(*widget: QWidget*)

Remove a widget from the title widget.

**Parameters**
**widget** – The widget to remove.

**ribbonHeight**() → int

Get the total height of the ribbon.

**Returns**
The height of the ribbon.

**ribbonVisible**() → bool

Get the visibility of the ribbon.

**Returns**
True if the ribbon is visible, False otherwise.

**rightToolBar**() → QToolBar

Return the right toolbar of the ribbon.

**Returns**
The right toolbar of the ribbon.

**setActiveAction**(*self*, *action: QAction | None*)

**setApplicationIcon**(*icon: QIcon*)

Set the application icon.

**Parameters**
**icon** – The icon to set.

**setAutoHideRibbon**(*autoHide: bool*)

Set whether the ribbon bar is automatically hidden when the mouse is pressed outside the ribbon bar.

**Parameters**
**autoHide** – Whether the ribbon bar is automatically hidden.

**setCollapseButtonIcon**(*icon: QIcon*)

Set the icon of the min button.

**Parameters**
**icon** – The icon to set.

**setCornerWidget**(*self*, *widget: QWidget | None*, *corner: Corner = Qt.TopRightCorner*)

**setCurrentCategory**(*category:* RibbonCategory)

Set the current category.

**Parameters**
**category** – The category to set.

**setDefaultUp**(*self*, *a0: bool*)

**setHelpButtonIcon**(*icon: QIcon*)

Set the icon of the help button.

**Parameters**
**icon** – The icon to set.

**setNativeMenuBar**(*self*, *nativeMenuBar: bool*)

**setQuickAccessButtonHeight**(*height: int = 30*)

Set the height of the quick access buttons.

**Parameters**
**height** – The height to set.

**setRibbonHeight**(*height: int*)

Set the total height of the ribbon.

**Parameters**
**height** – The height to set.

**setRibbonStyle**(*style:* RibbonStyle)

Set the style of the ribbon.

**Parameters**
**style** – The style to set.

**setRibbonVisible**(*visible: bool*)

Set the visibility of the ribbon.

**Parameters**
**visible** – True to show the ribbon, False to hide it.

**setRightToolBarHeight**(*height: int = 24*)

Set the height of the right buttons.

**Parameters**
**height** – The height to set.

**setTitle**(*title: str*)

Set the title of the ribbon.

**Parameters**
**title** – The title to set.

**showCategoryByIndex**(*index: int*)

Show category by tab index

**Parameters**
**index** – tab index

**showContextCategory**(*category:* RibbonContextCategory | RibbonContextCategories)

Show the given category or categories, if it is not a context category, nothing happens.

**Parameters**
**category** – The category to show.

**showRibbon**()

Show the ribbon.

**tabBar**() → *RibbonTabBar*

Return the tab bar of the ribbon.

**Returns**
The tab bar of the ribbon.

**title**() → str

Return the title of the ribbon.

**Returns**
The title of the ribbon.

class pyqtribbon.ribbonbar.**RibbonStackedWidget**(*parent=None*)

Bases: QStackedWidget

Stacked widget that is used to display the ribbon.

Create a new ribbon stacked widget.

**Parameters**
**parent** – The parent widget.

### pyqtribbon.screenshotwindow module

class pyqtribbon.screenshotwindow.**RibbonScreenShotWindow**(*fileName: str = 'shot.jpg', *args, **kwargs*)

Bases: QMainWindow

This class is just for taking a screenshot of the window, the window will be closed 0.1s after it is shown.

Initialize the class.

**Parameters**
**fileName** – The file name for the screenshot.

**setScreenShotFileName**(*fileName: str*)

Set the file name for the screenshot.

**Parameters**
**fileName** – The file name for the screenshot.

**takeScreenShot**()

Take a screenshot of the window.

### pyqtribbon.separator module

class pyqtribbon.separator.**RibbonHorizontalSeparator**(*width: int = 6*, *parent=None*)

Bases: *RibbonSeparator*

Horizontal separator.

Create a new horizontal separator.

**Parameters**

- **width** – The width of the separator.
- **parent** – The parent widget.

class pyqtribbon.separator.**RibbonSeparator**(*orientation=QtCore.Qt.Vertical*, *width=6*, *parent=None*)

class pyqtribbon.separator.**RibbonSeparator**(*parent=None*)

Bases: QFrame

The RibbonSeparator is a separator that can be used to separate widgets in a ribbon.

Create a new separator.

**Parameters**

- **orientation** – The orientation of the separator.
- **width** – The width of the separator.
- **parent** – The parent widget.

**paintEvent**(*event: QPaintEvent*) → None

Paint the separator.

**setTopBottomMargins**(*top: int*, *bottom: int*) → None

Set the top and bottom margins.

**sizeHint**() → QSize

Return the size hint.

class pyqtribbon.separator.**RibbonVerticalSeparator**(*width: int = 6*, *parent=None*)

Bases: *RibbonSeparator*

Vertical separator.

Create a new vertical separator.

**Parameters**

- **width** – The width of the separator.
- **parent** – The parent widget.

### pyqtribbon.tabbar module

**class** pyqtribbon.tabbar.**RibbonTabBar**(*parent=None*)

Bases: QTabBar

The TabBar for the title widget.

Create a new tab bar.

**Parameters**
**parent** – The parent widget.

**addAssociatedTabs**(*name: str*, *texts: List[str]*, *color: QColor*) → List[int]

Add associated multiple tabs which have the same color to the tab bar.

**Parameters**

- **name** – The name of the context category.
- **texts** – The texts of the tabs.
- **color** – The color of the tabs.

**Returns**
The indices of the tabs.

**addTab**(*text: str*, *color: QColor | None = None*, **args*, ***kwargs*) → int

Add a new tab to the tab bar.

**Parameters**

- **text** – The text of the tab.
- **color** – The color of the tab.

**Returns**
The index of the tab.

**changeColor**(*inx: int*) → None

Change tab's color.

**currentTabColor**() → QColor

Current tab color

**Returns**
Current tab color

**indexOf**(*tabName: str*) → int

Return the index of the tab with the given name.

**Parameters**
**tabName** – The name of the tab.

**Returns**
The index of the tab.

**removeAssociatedTabs**(*titles: List[str]*) → None

Remove tabs with the given titles.

**Parameters**
**titles** – The titles of the tabs to remove.

**tabTitles**() → List[str]

Return the titles of all tabs.

**Returns**
The titles of all tabs.

### pyqtribbon.titlewidget module

**class** pyqtribbon.titlewidget.**RibbonApplicationButton**

Bases: QToolButton

Application button in the ribbon bar.

**addFileMenu**() → *RibbonMenu*

Add a new ribbon menu to the application button.

**Returns**
The new ribbon menu.

**class** pyqtribbon.titlewidget.**RibbonTitleLabel**

Bases: QLabel

Title label in the ribbon bar.

**class** pyqtribbon.titlewidget.**RibbonTitleWidget**(*title='PyQtRibbon'*, *parent=None*)

**class** pyqtribbon.titlewidget.**RibbonTitleWidget**(*parent=None*)

Bases: QFrame

The title widget of the ribbon.

Initialize the ribbon title widget.

**Parameters**

- **title** – The title of the ribbon.
- **parent** – The parent widget.

**addQuickAccessButton**(*button: QToolButton*)

Add a widget to the quick access bar.

**Parameters**
**button** – The button to add.

**addRightToolButton**(*button: QToolButton*)

Add a widget to the right button bar.

**Parameters**
**button** – The button to add.

**addTitleWidget**(*widget: QWidget*)

Add a widget to the title layout.

**Parameters**
**widget** – The widget to add.

**applicationButton**() → *RibbonApplicationButton*

Return the application button.

**collapseRibbonButton**() → QToolButton

Return the collapse ribbon button.

**Returns**
The collapse ribbon button.

**collapseRibbonButtonClicked**(*bool*)

Signal, the collapse button wa clicked.

**helpButtonClicked**(*bool*)

Signal, the help button was clicked.

**helpRibbonButton**() → QToolButton

Return the help ribbon button.

**Returns**
The help ribbon button.

**insertTitleWidget**(*index: int*, *widget: QWidget*)

Insert a widget to the title layout.

**Parameters**

- **index** – The index to insert the widget.
- **widget** – The widget to insert.

**mouseDoubleClickEvent**(*self*, *a0: QMouseEvent | None*)

**mouseMoveEvent**(*self*, *a0: QMouseEvent | None*)

**mousePressEvent**(*self*, *a0: QMouseEvent | None*)

**quickAccessButtons**() → List[QToolButton]

Return the quick access buttons of the ribbon.

**Returns**
The quick access buttons of the ribbon.

**quickAccessToolBar**() → QToolBar

Return the quick access toolbar of the ribbon.

**Returns**
The quick access toolbar of the ribbon.

**removeCollapseButton**()

Remove the min button from the ribbon.

**removeHelpButton**()

Remove the help button from the ribbon.

**removeTitleWidget**(*widget: QWidget*)

Remove a widget from the title layout.

**Parameters**
**widget** – The widget to remove.

**rightToolBar**() → QToolBar

Return the right toolbar of the ribbon.

**Returns**
The right toolbar of the ribbon.

**setApplicationIcon**(*icon: QIcon*)

Set the application icon.

**Parameters**
**icon** – The icon to set.

**setCollapseButtonIcon**(*icon: QIcon*)

Set the icon of the min button.

**Parameters**
**icon** – The icon to set.

**setHelpButtonIcon**(*icon: QIcon*)

Set the icon of the help button.

**Parameters**
**icon** – The icon to set.

**setQuickAccessButtonHeight**(*height: int = 30*)

Set the height of the quick access buttons.

**Parameters**
**height** – The height to set.

**setRightToolBarHeight**(*height: int = 24*)

Set the height of the right buttons.

**Parameters**
**height** – The height to set.

**setTitle**(*title: str*)

Set the title of the ribbon.

**Parameters**
**title** – The title to set.

**tabBar**() → *RibbonTabBar*

Return the tab bar of the ribbon.

**Returns**
The tab bar of the ribbon.

**title**() → str

Return the title of the ribbon.

**Returns**
The title of the ribbon.

**topLevelWidget**() → QWidget

### pyqtribbon.toolbutton module

class pyqtribbon.toolbutton.**RibbonToolButton**(*parent=None*)

Bases: QToolButton

Tool button that is showed in the ribbon.

Create a new ribbon tool button.

**Parameters**
**parent** – The parent widget.

**addRibbonMenu**() → *RibbonMenu*

Add a ribbon menu for the button.

**Returns**
The added ribbon menu.

**buttonStyle**() → *RibbonButtonStyle*

Get the button style of the button.

**Returns**
The button style of the button.

**maximumIconSize**() → int

Get the maximum icon size of the button.

**Returns**
The maximum icon size of the button.

**setButtonStyle**(*style:* RibbonButtonStyle)

Set the button style of the button.

**Parameters**
**style** – The button style of the button.

**setMaximumIconSize**(*size: int*)

Set the maximum icon size of the button.

**Parameters**
**size** – The maximum icon size of the button.

### pyqtribbon.utils module

pyqtribbon.utils.**DataFile**(*filename*)

Return the path to a data file.

**Parameters**
**filename** – The filename of the data file.

**Returns**
The path to the data file.

pyqtribbon.version module

### 4.1.2 Module contents

CHAPTER

FIVE

# INDICES AND TABLES

- genindex
- modindex
- search

# PYTHON MODULE INDEX

## p

pyqtribbon, 48
pyqtribbon.category, 19
pyqtribbon.constants, 24
pyqtribbon.gallery, 25
pyqtribbon.logger, 27
pyqtribbon.menu, 27
pyqtribbon.panel, 28
pyqtribbon.ribbonbar, 35
pyqtribbon.screenshotwindow, 41
pyqtribbon.separator, 42
pyqtribbon.tabbar, 43
pyqtribbon.titlewidget, 44
pyqtribbon.toolbutton, 47
pyqtribbon.utils, 47
pyqtribbon.version, 48

# INDEX

## A


actionAdded (*pyqtribbon.menu.RibbonPermanentMenu attribute*), 28
actionAt() (*pyqtribbon.ribbonbar.RibbonBar method*), 35
actionGeometry() (*pyqtribbon.ribbonbar.RibbonBar method*), 35
activeAction() (*pyqtribbon.ribbonbar.RibbonBar method*), 35
addAction() (*pyqtribbon.menu.RibbonPermanentMenu method*), 28
addAction() (*pyqtribbon.ribbonbar.RibbonBar method*), 35
addAssociatedTabs() (*pyqtribbon.tabbar.RibbonTabBar method*), 43
addButton() (*pyqtribbon.gallery.RibbonGallery method*), 25
addButton() (*pyqtribbon.panel.RibbonPanel method*), 29
addCalendarWidget() (*pyqtribbon.panel.RibbonPanel method*), 29
addCategoriesBy() (*pyqtribbon.ribbonbar.RibbonBar method*), 35
addCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 36
addComboBox() (*pyqtribbon.panel.RibbonPanel method*), 29
addContextCategories() (*pyqtribbon.ribbonbar.RibbonBar method*), 36
addContextCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 36
addDateEdit() (*pyqtribbon.panel.RibbonPanel method*), 30
addDateTimeEdit() (*pyqtribbon.panel.RibbonPanel method*), 30
addDoubleSpinBox() (*pyqtribbon.panel.RibbonPanel method*), 30
addFileMenu() (*pyqtribbon.ribbonbar.RibbonBar method*), 36
addFileMenu() (*pyqtribbon.titlewidget.RibbonApplicationButton method*), 44
addFontComboBox() (*pyqtribbon.panel.RibbonPanel method*), 30
addFormLayoutWidget() (*pyqtribbon.menu.RibbonMenu method*), 27
addGallery() (*pyqtribbon.panel.RibbonPanel method*), 30
addGridLayoutWidget() (*pyqtribbon.menu.RibbonMenu method*), 27
addHorizontalLayoutWidget() (*pyqtribbon.menu.RibbonMenu method*), 27
addHorizontalSeparator() (*pyqtribbon.panel.RibbonPanel method*), 30
addLabel() (*pyqtribbon.menu.RibbonMenu method*), 27
addLabel() (*pyqtribbon.panel.RibbonPanel method*), 30
addLargeButton() (*pyqtribbon.panel.RibbonPanel method*), 30
addLargeToggleButton() (*pyqtribbon.panel.RibbonPanel method*), 30
addLargeWidget() (*pyqtribbon.panel.RibbonPanel method*), 30
addLineEdit() (*pyqtribbon.panel.RibbonPanel method*), 30
addListWidget() (*pyqtribbon.panel.RibbonPanel method*), 30
addMediumButton() (*pyqtribbon.panel.RibbonPanel method*), 30
addMediumToggleButton() (*pyqtribbon.panel.RibbonPanel method*), 30
addMediumWidget() (*pyqtribbon.panel.RibbonPanel method*), 30
addMenu() (*pyqtribbon.ribbonbar.RibbonBar method*), 36
addNormalCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 36
addPanel() (*pyqtribbon.category.RibbonCategory method*), 19
addPanelsBy() (*pyqtribbon.category.RibbonCategory method*), 19
addPlainTextEdit() (*pyqtribbon.panel.RibbonPanel method*), 31
addProgressBar() (*pyqtribbon.panel.RibbonPanel method*), 31

addQuickAccessButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
addQuickAccessButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 44
addRibbonMenu() (*pyqtribbon.toolbutton.RibbonToolButton method*), 47
addRightToolButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
addRightToolButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 44
addSeparator() (*pyqtribbon.panel.RibbonPanel method*), 31
addSeparator() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
addSlider() (*pyqtribbon.panel.RibbonPanel method*), 31
addSmallButton() (*pyqtribbon.panel.RibbonPanel method*), 31
addSmallToggleButton() (*pyqtribbon.panel.RibbonPanel method*), 31
addSmallWidget() (*pyqtribbon.panel.RibbonPanel method*), 31
addSpacing() (*pyqtribbon.menu.RibbonMenu method*), 27
addSpinBox() (*pyqtribbon.panel.RibbonPanel method*), 31
addTab() (*pyqtribbon.tabbar.RibbonTabBar method*), 43
addTableWidget() (*pyqtribbon.panel.RibbonPanel method*), 31
addTextEdit() (*pyqtribbon.panel.RibbonPanel method*), 31
addTimeEdit() (*pyqtribbon.panel.RibbonPanel method*), 31
addTitleWidget() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
addTitleWidget() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 44
addToggleButton() (*pyqtribbon.gallery.RibbonGallery method*), 25
addToggleButton() (*pyqtribbon.panel.RibbonPanel method*), 31
addTreeWidget() (*pyqtribbon.panel.RibbonPanel method*), 31
addVerticalLayoutWidget() (*pyqtribbon.menu.RibbonMenu method*), 27
addVerticalSeparator() (*pyqtribbon.panel.RibbonPanel method*), 31
addWidget() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 21
addWidget() (*pyqtribbon.menu.RibbonMenu method*), 28
addWidget() (*pyqtribbon.panel.RibbonPanel method*), 31
addWidget() (*pyqtribbon.panel.RibbonPanelItemWidget method*), 34
addWidgetsBy() (*pyqtribbon.panel.RibbonPanel method*), 32
applicationButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 44
applicationOptionButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
autoSetScrollButtonsVisible() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 21

## B

buttonStyle() (*pyqtribbon.toolbutton.RibbonToolButton method*), 47

## C

categories() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
categoriesVisible() (*pyqtribbon.category.RibbonContextCategories method*), 22
category() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
categoryStyle() (*pyqtribbon.category.RibbonCategory method*), 20
categoryVisible() (*pyqtribbon.category.RibbonContextCategory method*), 23
categoryVisible() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
changeColor() (*pyqtribbon.tabbar.RibbonTabBar method*), 43
clear() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
collapseRibbonButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 37
collapseRibbonButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 44
collapseRibbonButtonClicked (*pyqtribbon.titlewidget.RibbonTitleWidget attribute*), 45
color() (*pyqtribbon.category.RibbonContextCategories method*), 22
color() (*pyqtribbon.category.RibbonContextCategory method*), 23

ColumnWise (*pyqtribbon.constants.RibbonSpaceFindMode attribute*), 24
Context (*pyqtribbon.constants.RibbonCategoryStyle attribute*), 24
contextColors (*in module pyqtribbon.constants*), 24
cornerWidget() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
currentCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
currentTabColor() (*pyqtribbon.tabbar.RibbonTabBar method*), 43

## D

DataFile() (*in module pyqtribbon.utils*), 47
Debug (*pyqtribbon.constants.RibbonStyle attribute*), 24
Default (*pyqtribbon.constants.RibbonStyle attribute*), 24
defaultRowSpan() (*pyqtribbon.panel.RibbonPanel method*), 32
displayOptionsButtonClicked (*pyqtribbon.category.RibbonCategoryLayoutWidget attribute*), 21

## E

eventFilter() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
exception_hook() (*pyqtribbon.logger.UncaughtHook method*), 27

## H

helpButtonClicked (*pyqtribbon.ribbonbar.RibbonBar attribute*), 38
helpButtonClicked (*pyqtribbon.titlewidget.RibbonTitleWidget attribute*), 45
helpRibbonButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
helpRibbonButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
hideContextCategories() (*pyqtribbon.category.RibbonContextCategories method*), 22
hideContextCategory() (*pyqtribbon.category.RibbonContextCategory method*), 23
hideContextCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
hideEvent() (*pyqtribbon.menu.RibbonPermanentMenu method*), 28
hidePopupWidget() (*pyqtribbon.gallery.RibbonGallery method*), 25
hideRibbon() (*pyqtribbon.ribbonbar.RibbonBar method*), 38

## I

indexOf() (*pyqtribbon.tabbar.RibbonTabBar method*), 43
insertMenu() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
insertSeparator() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
insertTitleWidget() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
insertTitleWidget() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
isDefaultUp() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
isNativeMenuBar() (*pyqtribbon.ribbonbar.RibbonBar method*), 38

## L

Large (*pyqtribbon.constants.RibbonButtonStyle attribute*), 24
largeRows() (*pyqtribbon.panel.RibbonPanel method*), 32

## M

maximumIconSize() (*pyqtribbon.toolbutton.RibbonToolButton method*), 47
maximumRows() (*pyqtribbon.panel.RibbonPanel method*), 32
Medium (*pyqtribbon.constants.RibbonButtonStyle attribute*), 24
mediumRows() (*pyqtribbon.panel.RibbonPanel method*), 33
minimumSizeHint() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
module
    pyqtribbon, 48
    pyqtribbon.category, 19
    pyqtribbon.constants, 24
    pyqtribbon.gallery, 25
    pyqtribbon.logger, 27
    pyqtribbon.menu, 27
    pyqtribbon.panel, 28
    pyqtribbon.ribbonbar, 35
    pyqtribbon.screenshotwindow, 41
    pyqtribbon.separator, 42
    pyqtribbon.tabbar, 43
    pyqtribbon.titlewidget, 44
    pyqtribbon.toolbutton, 47
    pyqtribbon.utils, 47
    pyqtribbon.version, 48
mouseDoubleClickEvent() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45

mouseMoveEvent() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
mousePressEvent() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45

## N

name() (*pyqtribbon.category.RibbonContextCategories method*), 22
Normal (*pyqtribbon.constants.RibbonCategoryStyle attribute*), 24

## P

paintEvent() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 21
paintEvent() (*pyqtribbon.separator.RibbonSeparator method*), 42
panel() (*pyqtribbon.category.RibbonCategory method*), 20
panelOptionButton() (*pyqtribbon.panel.RibbonPanel method*), 33
panelOptionClicked (*pyqtribbon.panel.RibbonPanel attribute*), 33
panels() (*pyqtribbon.category.RibbonCategory method*), 20
popupMenu() (*pyqtribbon.gallery.RibbonGallery method*), 25
popupWindowSize() (*pyqtribbon.gallery.RibbonGallery method*), 26
pyqtribbon
    module, 48
pyqtribbon.category
    module, 19
pyqtribbon.constants
    module, 24
pyqtribbon.gallery
    module, 25
pyqtribbon.logger
    module, 27
pyqtribbon.menu
    module, 27
pyqtribbon.panel
    module, 28
pyqtribbon.ribbonbar
    module, 35
pyqtribbon.screenshotwindow
    module, 41
pyqtribbon.separator
    module, 42
pyqtribbon.tabbar
    module, 43
pyqtribbon.titlewidget
    module, 44
pyqtribbon.toolbutton
    module, 47
pyqtribbon.utils
    module, 47
pyqtribbon.version
    module, 48

## Q

quickAccessButtons() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
quickAccessToolBar() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
quickAccessToolBar() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45

## R

removeAssociatedTabs() (*pyqtribbon.tabbar.RibbonTabBar method*), 43
removeCategories() (*pyqtribbon.ribbonbar.RibbonBar method*), 38
removeCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
removeCollapseButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
removeCollapseButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
removeHelpButton() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
removeHelpButton() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
removePanel() (*pyqtribbon.category.RibbonCategory method*), 20
removeTitleWidget() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
removeTitleWidget() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
removeWidget() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 21
removeWidget() (*pyqtribbon.panel.RibbonPanel method*), 33
request_cells() (*pyqtribbon.panel.RibbonGridLayoutManager method*), 28
resizeEvent() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 21

resizeEvent() (*pyqtribbon.gallery.RibbonGallery method*), 26
resizeEvent() (*pyqtribbon.gallery.RibbonGalleryListWidget method*), 26
RibbonApplicationButton (*class in pyqtribbon.titlewidget*), 44
ribbonArguments (*pyqtribbon.panel.RibbonPanel attribute*), 33
RibbonBar (*class in pyqtribbon.ribbonbar*), 35
RibbonButtonStyle (*class in pyqtribbon.constants*), 24
RibbonCategory (*class in pyqtribbon.category*), 19
RibbonCategoryLayoutButton (*class in pyqtribbon.category*), 21
RibbonCategoryLayoutWidget (*class in pyqtribbon.category*), 21
RibbonCategoryScrollArea (*class in pyqtribbon.category*), 22
RibbonCategoryScrollAreaContents (*class in pyqtribbon.category*), 22
RibbonCategoryStyle (*class in pyqtribbon.constants*), 24
RibbonContextCategories (*class in pyqtribbon.category*), 22
RibbonContextCategory (*class in pyqtribbon.category*), 22
RibbonGallery (*class in pyqtribbon.gallery*), 25
RibbonGalleryButton (*class in pyqtribbon.gallery*), 26
RibbonGalleryListWidget (*class in pyqtribbon.gallery*), 26
RibbonGalleryPopupListWidget (*class in pyqtribbon.gallery*), 26
RibbonGridLayoutManager (*class in pyqtribbon.panel*), 28
ribbonHeight() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
RibbonHorizontalSeparator (*class in pyqtribbon.separator*), 42
RibbonMenu (*class in pyqtribbon.menu*), 27
RibbonNormalCategory (*class in pyqtribbon.category*), 23
RibbonPanel (*class in pyqtribbon.panel*), 29
RibbonPanelItemWidget (*class in pyqtribbon.panel*), 34
RibbonPanelOptionButton (*class in pyqtribbon.panel*), 34
RibbonPanelTitle (*class in pyqtribbon.panel*), 35
RibbonPermanentMenu (*class in pyqtribbon.menu*), 28
RibbonPopupWidget (*class in pyqtribbon.gallery*), 26
RibbonScreenShotWindow (*class in pyqtribbon.screenshotwindow*), 41
RibbonSeparator (*class in pyqtribbon.separator*), 42
RibbonSpaceFindMode (*class in pyqtribbon.constants*), 24
RibbonStackedWidget (*class in pyqtribbon.ribbonbar*), 41
RibbonStyle (*class in pyqtribbon.constants*), 24
RibbonTabBar (*class in pyqtribbon.tabbar*), 43
RibbonTitleLabel (*class in pyqtribbon.titlewidget*), 44
RibbonTitleWidget (*class in pyqtribbon.titlewidget*), 44
RibbonToolButton (*class in pyqtribbon.toolbutton*), 47
RibbonVerticalSeparator (*class in pyqtribbon.separator*), 42
ribbonVisible() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
rightToolBar() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
rightToolBar() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
rowHeight() (*pyqtribbon.panel.RibbonPanel method*), 33
RowWise (*pyqtribbon.constants.RibbonSpaceFindMode attribute*), 24

## S

scrollNext() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 22
scrollPrevious() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 22
scrollToNextRow() (*pyqtribbon.gallery.RibbonGalleryListWidget method*), 26
scrollToPreviousRow() (*pyqtribbon.gallery.RibbonGalleryListWidget method*), 26
setActiveAction() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
setApplicationIcon() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
setApplicationIcon() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 45
setAutoHideRibbon() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
setButtonStyle() (*pyqtribbon.toolbutton.RibbonToolButton method*), 47
setCategoriesVisible() (*pyqtribbon.category.RibbonContextCategories method*), 22
setCategoryStyle() (*pyqtribbon.category.RibbonCategory method*), 20

setCategoryStyle() (*pyqtribbon.category.RibbonContextCategory method*), 23
setCategoryStyle() (*pyqtribbon.category.RibbonNormalCategory method*), 23
setCategoryVisible() (*pyqtribbon.category.RibbonContextCategory method*), 23
setCollapseButtonIcon() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
setCollapseButtonIcon() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
setColor() (*pyqtribbon.category.RibbonContextCategories method*), 22
setColor() (*pyqtribbon.category.RibbonContextCategory method*), 23
setCornerWidget() (*pyqtribbon.ribbonbar.RibbonBar method*), 39
setCurrentCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setDefaultUp() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setHelpButtonIcon() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setHelpButtonIcon() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
setLargeRows() (*pyqtribbon.panel.RibbonPanel method*), 33
setMaximumIconSize() (*pyqtribbon.toolbutton.RibbonToolButton method*), 47
setMaximumRows() (*pyqtribbon.category.RibbonCategory method*), 20
setMaximumRows() (*pyqtribbon.panel.RibbonPanel method*), 33
setMediumRows() (*pyqtribbon.panel.RibbonPanel method*), 33
setName() (*pyqtribbon.category.RibbonContextCategories method*), 22
setNativeMenuBar() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setPanelOptionToolTip() (*pyqtribbon.panel.RibbonPanel method*), 33
setPopupHideOnClick() (*pyqtribbon.gallery.RibbonGallery method*), 26
setPopupWindowSize() (*pyqtribbon.gallery.RibbonGallery method*), 26
setQuickAccessButtonHeight() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setQuickAccessButtonHeight() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
setRibbonHeight() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setRibbonStyle() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setRibbonVisible() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setRightToolBarHeight() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setRightToolBarHeight() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
setScreenShotFileName() (*pyqtribbon.screenshotwindow.RibbonScreenShotWindow method*), 41
setSelectedButton() (*pyqtribbon.gallery.RibbonGallery method*), 26
setSmallRows() (*pyqtribbon.panel.RibbonPanel method*), 34
setTitle() (*pyqtribbon.panel.RibbonPanel method*), 34
setTitle() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
setTitle() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
setTopBottomMargins() (*pyqtribbon.separator.RibbonSeparator method*), 42
show_exception_box() (*pyqtribbon.logger.UncaughtHook static method*), 27
showCategoryByIndex() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
showContextCategories() (*pyqtribbon.category.RibbonContextCategories method*), 22
showContextCategory() (*pyqtribbon.category.RibbonContextCategory method*), 23
showContextCategory() (*pyqtribbon.ribbonbar.RibbonBar method*), 40
showPopup() (*pyqtribbon.gallery.RibbonGallery method*), 26
showRibbon() (*pyqtribbon.ribbonbar.RibbonBar method*), 41
sizeHint() (*pyqtribbon.separator.RibbonSeparator method*), 42
Small (*pyqtribbon.constants.RibbonButtonStyle attribute*), 24
smallRows() (*pyqtribbon.panel.RibbonPanel method*), 34

## T

tabBar() (*pyqtribbon.ribbonbar.RibbonBar method*), 41

tabBar() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
tabTitles() (*pyqtribbon.tabbar.RibbonTabBar method*), 43
takePanel() (*pyqtribbon.category.RibbonCategory method*), 20
takeScreenShot() (*pyqtribbon.screenshotwindow.RibbonScreenShotWindow method*), 41
takeWidget() (*pyqtribbon.category.RibbonCategoryLayoutWidget method*), 22
title() (*pyqtribbon.category.RibbonCategory method*), 21
title() (*pyqtribbon.panel.RibbonPanel method*), 34
title() (*pyqtribbon.ribbonbar.RibbonBar method*), 41
title() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46
topLevelWidget() (*pyqtribbon.titlewidget.RibbonTitleWidget method*), 46

## U

UncaughtHook (*class in pyqtribbon.logger*), 27

## W

widget() (*pyqtribbon.panel.RibbonPanel method*), 34
widgets() (*pyqtribbon.panel.RibbonPanel method*), 34