FREE eBook

LEARNING pyqt4

Free unaffiliated eBook created from
Stack Overflow contributors.

#pyqt4

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: pyqt4

It is an unofficial and free pyqt4 ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official pyqt4.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with pyqt4

Remarks

This section provides an overview of what `PyQt4` is, and why a developer might want to use it.

PyQt is a GUI widgets toolkit. It is a Python interface for Qt, one of the most powerful, and popular cross-platform GUI library. PyQt is a blend of Python programming language and the Qt library. This introductory tutorial will assist you in creating graphical applications with the help of PyQt.

It should also mention any large subjects within `PyQt4`, and link out to the related topics. Since the Documentation for `PyQt4` is new, you may need to create initial versions of those related topics.

Examples

Installation or Setup

Detailed instructions on getting pyqt4 set up or installed.

1. Windows Link
2. Mac Link
3. Linux Link

If you want to install the version specific to your system python version and you system configuration(32-bit or 64-bit) then go to this link and download and install the package. You can install the `.whl` file simply by going to command prompt `pip install PyQt4-4.11.4-cp34-none-win_amd64.whl`.

Go a head Install the software and Start Building Awesome GUI!!

Basic Hello World Program

```
import sys
from PyQt4 import QtGui

def window():
   app = QtGui.QApplication(sys.argv)
   w = QtGui.QWidget()
   b = QtGui.QLabel(w)
   b.setText("<h1>Welcome to PyQt4 SO Documentation!</h1>")
   w.setGeometry(100,100,550,65)
   b.move(50,20)
   w.setWindowTitle("PyQt4 Hello World Demo")
   w.show()
   sys.exit(app.exec_())

if __name__ == '__main__':
   window()
```

Read Getting started with pyqt4 online: https://riptutorial.com/pyqt4/topic/8665/getting-started-with-pyqt4

Chapter 2: Basic Widgets : QLabel

Introduction

`QLabel` is used for displaying text or an image. No user interaction functionality is provided. The visual appearance of the label can be configured in various ways, and it can be used for specifying a focus mnemonic key for another widget.

Remarks

A `QLabel` object acts as a placeholder to display *non-editable* **text**,**image**, or a **animated GIF**. It can also be used as a mnemonic key for other widgets.

In this example, `QLabel` objects `l2` and `l4` have the caption containing hyperlink. `setOpenExternalLinks` for `l2` is set to **true**. Hence, if this label is **clicked**, the associated URL will open in the browser. `linkHovered` *signal* of `l4` is connected to `hovered()` function. So, whenever the mouse hovers over it, the function will be executed.

QPixmap object prepares offscreen image from *python.jpg* file. It is displayed as label `l3` by using `setPixmap()` method

Plain text, **hyperlink** or **rich text** can also be displayed in `QLabel`.

In `QLabel` most of the basic HTML tags are allowed. For Example : `h1`,`h2`,`h3`,`font`,`span`,etc

You can visit here for the list of supported HTML subset.

Examples

QLabel example with text, hyperlink and image

Following is the example of `QLabel` that displays use of texts,images and hyperlinks.

```
import sys
from PyQt4.QtCore import *
from PyQt4.QtGui import *

def window():
   app = QApplication(sys.argv)
   win = QWidget()

   l1 = QLabel()
   l2 = QLabel()
   l3 = QLabel()
   l4 = QLabel()

   l1.setText("<h1>Hello World</h1>")
   l4.setText("<b>Hello Stack OverFlow</b>")
   l2.setText("<font color='red'>Welcome To Stack Overflow Documentation</font>")
```

```
    l1.setAlignment(Qt.AlignCenter)
    l3.setAlignment(Qt.AlignCenter)
    l4.setAlignment(Qt.AlignRight)
    l3.setPixmap(QPixmap("python.jpg"))

    vbox = QVBoxLayout()
    vbox.addWidget(l1)
    vbox.addStretch()
    vbox.addWidget(l2)
    vbox.addStretch()
    vbox.addWidget(l3)
    vbox.addStretch()
    vbox.addWidget(l4)

    l1.setOpenExternalLinks(True)
    l4.linkActivated.connect(clicked)
    l2.linkHovered.connect(hovered)
    l1.setTextInteractionFlags(Qt.TextSelectableByMouse)
    win.setLayout(vbox)

    win.setWindowTitle("PyQt4 QLabel Demo")
    win.show()
    sys.exit(app.exec_())

def hovered():
    print ("Come'On Click ME")
def clicked():
    print ("You Clicked Me")

if __name__ == '__main__':
    window()
```

The code outputs the following result:

Read Basic Widgets : QLabel online: https://riptutorial.com/pyqt4/topic/9312/basic-widgets---qlabel

Chapter 3: Hello World Program

Introduction

You need to know basics of Python Programming Language before you start with PyQt.

PyQt is a GUI widgets toolkit. It is a Python interface for Qt, one of the most powerful, and popular cross-platform GUI library. PyQt is a blend of Python programming language and the Qt library.

Here is an Hello World Program to get you started.

Remarks

Creating a simple GUI application using PyQt involves the following steps –

1. Import QtGui module
2. Create an application object.
3. A QWidget object creates top level window. Add QLabel object in it.
4. Set the caption of label as “hello world”.
5. Define the size and position of window by setGeometry() method.
6. Enter the mainloop of application by app.exec_() method.

Examples

Hello World Program

```
import sys
from PyQt4 import QtGui

def window():
   app = QtGui.QApplication(sys.argv)
   w = QtGui.QWidget()
   b = QtGui.QLabel(w)
   b.setText("Hello World!")
   w.setGeometry(100,100,200,50)
   b.move(50,20)
   w.setWindowTitle("PyQt")
   w.show()
   sys.exit(app.exec_())

if __name__ == '__main__':
   window()
```

Read Hello World Program online: https://riptutorial.com/pyqt4/topic/9247/hello-world-program

Chapter 4: QMessageBox

Introduction

`QMessageBox` is the simplest way to give (or ask) an information to (or from) the user. It's a modal dialog, inheriting the `QDialog` class. It also has four convenience static functions: `information`, `question`, `warning` and `critical`.

Examples

Basic usage: Hello World

```
app = QApplication( sys.argv )
box = QMessageBox()

# Window Title
box.setWindowTitle( "Hello World." )

# Icon: Information, Warning, Question, Critical
box.setIcon( QMessageBox.Information )

# Short version of the information
box.setText( "Hello World!" )

# Informative text
box.setInformativeText( "Hello World! We are using Qt to display this beautiful dialog to
you." )

# Show the messagebox as a modal dialog
box.exec_()

return 0
```

Read QMessageBox online: https://riptutorial.com/pyqt4/topic/9806/qmessagebox

Chapter 5: Signals and Slots

Introduction

Functions or methods are executed in response to user's actions like `clicking` on a button, selecting an item from a collection or a mouse click etc., called **`events`**.

Each PyQt widget, which is derived from `QObject` class, is designed to emit **signal** in response to one or more events. The signal on its own does not perform any action. Instead, it is **connected** to a **slot**.

Remarks

In the following example, two `QPushButton` objects (`b1` and `b2`) are added in `QDialog` window. We want to call functions `b1_clicked()` and `b2_clicked()` on clicking `b1` and `b2` respectively.

When `b1` is clicked, the `clicked()` signal is connected to `b1_clicked()` function

```
b1.clicked.connect(b1_clicked())
```

When `b2` is clicked, the `clicked()` signal is connected to `b2_clicked()` function

```
QObject.connect(b2, SIGNAL("clicked()"), b2_clicked)
```

Widgets used to build the GUI interface act as the source of such events.

Each PyQt widget, which is derived from `QObject` class, is designed to emit **signal** in response to one or more events. The signal on its own does not perform any action. Instead, it is **connected** to a **slot**. The slot can be any **callable Python function**.

Examples

An Example Using Signals and Slots

```
import sys
from PyQt4.QtCore import *
from PyQt4.QtGui import *

def window():
   app = QApplication(sys.argv)
   win = QDialog()
   b1 = QPushButton(win)
   b1.setText("Button1")
   b1.move(50,20)
   b1.clicked.connect(b1_clicked)

   b2 = QPushButton(win)
```

```
   b2.setText("Button2")
   b2.move(50,50)
   QObject.connect(b2,SIGNAL("clicked()"),b2_clicked)

   win.setGeometry(100,100,200,100)
   win.setWindowTitle("PyQt")
   win.show()
   sys.exit(app.exec_())

def b1_clicked():
   print ("Button 1 clicked")

def b2_clicked():
   print ("Button 2 clicked")

if __name__ == '__main__':
   window()
```

The following example will produce two buttons, if you click them, then function `b1_clicked` or `b2_clicked` will be called.

Read Signals and Slots online: https://riptutorial.com/pyqt4/topic/9270/signals-and-slots

Credits

S. No	Chapters	Contributors
1	Getting started with pyqt4	Ahmad Taha, Community, learncode
2	Basic Widgets : QLabel	Ahmad Taha
3	Hello World Program	Ahmad Taha
4	QMessageBox	Marcus
5	Signals and Slots	Ahmad Taha