eBook Gratuit

APPRENEZ pyqt

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#pyqt

Table des matières

À propos

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

It is an unofficial and free pyqt 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 pyqt.

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

Chapitre 1: Démarrer avec pyqt

Remarques

PyQt est une liaison Python au populaire framework d'application Qt multiplate-forme couramment utilisé pour créer des applications graphiques. PyQt4 prend en charge Qt4 et PyQt5 prend en charge Qt5. Il fonctionne sur toutes les plates-formes supportées par Qt (Windows, OS X, Linux, iOS et Android). Les liaisons sont implémentées sous la forme d'un ensemble de modules et de classes Python.

Pour plus d'informations, consultez le site Web PyQt .

Examples

Installation de PyQt4

Méthode d'installation suggérée

Windows : Téléchargez et exécutez le fichier de configuration binaire .

Linux (Debian) : Exécutez cette commande dans votre ligne de commande:

```
$ apt-get install python-qt4 pyqt4-dev-tools qt4-designer
```

OS X : Exécutez cette commande dans votre ligne de commande:

```
$ brew install pyqt
```

Installer manuellement

Vous pouvez également télécharger le code source manuellement à partir de là , puis l'installer et le configurer vous-même.

Testez votre installation

Si pyqt est installé correctement, vous pourrez exécuter la commande `pyuic4` . S'il est installé correctement, vous verrez l'erreur suivante:

```
$ pyuic4
Error: one input ui-file must be specified
```

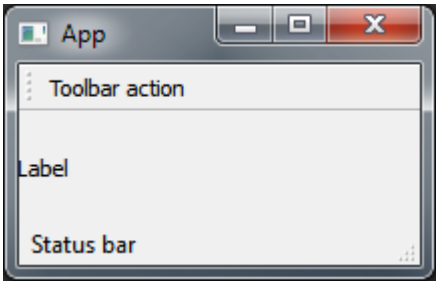
Installation complète

Vous avez maintenant installé la bibliothèque PyQt4. Deux applications utiles ont également été installées le long de PyQt4:

- Qt Designer: une application pour la conception 'drag & drop' d'interfaces graphiques (crée des fichiers `.ui`),
- pyuic4: une application en ligne de commande capable de convertir des fichiers `.ui` en code Python.

Une application de base

L'exemple suivant montre une fenêtre d'interface graphique principale de base avec un widget d'étiquette, une barre d'outils et une barre d'état utilisant PyQt4.




```
import sys
from PyQt4 import QtGui


class App(QtGui.QApplication):
    def __init__(self, sys_argv):
        super(App, self).__init__(sys_argv)
        self.build_ui()

    def build_ui(self):
        # build a main GUI window
        self.main_window = QtGui.QMainWindow()
        self.main_window.setWindowTitle('App')
        self.main_window.show()

        # add a label to the main window
        label = QtGui.QLabel('Label')
        self.main_window.setCentralWidget(label)

        # add a toolbar with an action button to the main window
        action = QtGui.QAction('Toolbar action', self)
        toolbar = QtGui.QToolBar()
        toolbar.addAction(action)
        self.main_window.addToolBar(toolbar)

        # add a status bar to the main window
        status_bar = QtGui.QStatusBar()
        status_bar.showMessage('Status bar')
        self.main_window.setStatusBar(status_bar)


if __name__ == '__main__':
    app = App(sys.argv)
    sys.exit(app.exec_())
```

Bonjour le monde

Ce code de base lancera une fenêtre graphique "Hello world" en utilisant PyQt4:

```
import sys
from PyQt4 import QtGui

# create instance of QApplication
app = QtGui.QApplication(sys.argv)

# create QLabel, without parent it will be shown as window
label = QtGui.QLabel('Hello world!')
label.show()

# start the execution loop of the application
sys.exit(app.exec_())
```

Ceci est le même code en utilisant PyQt5.

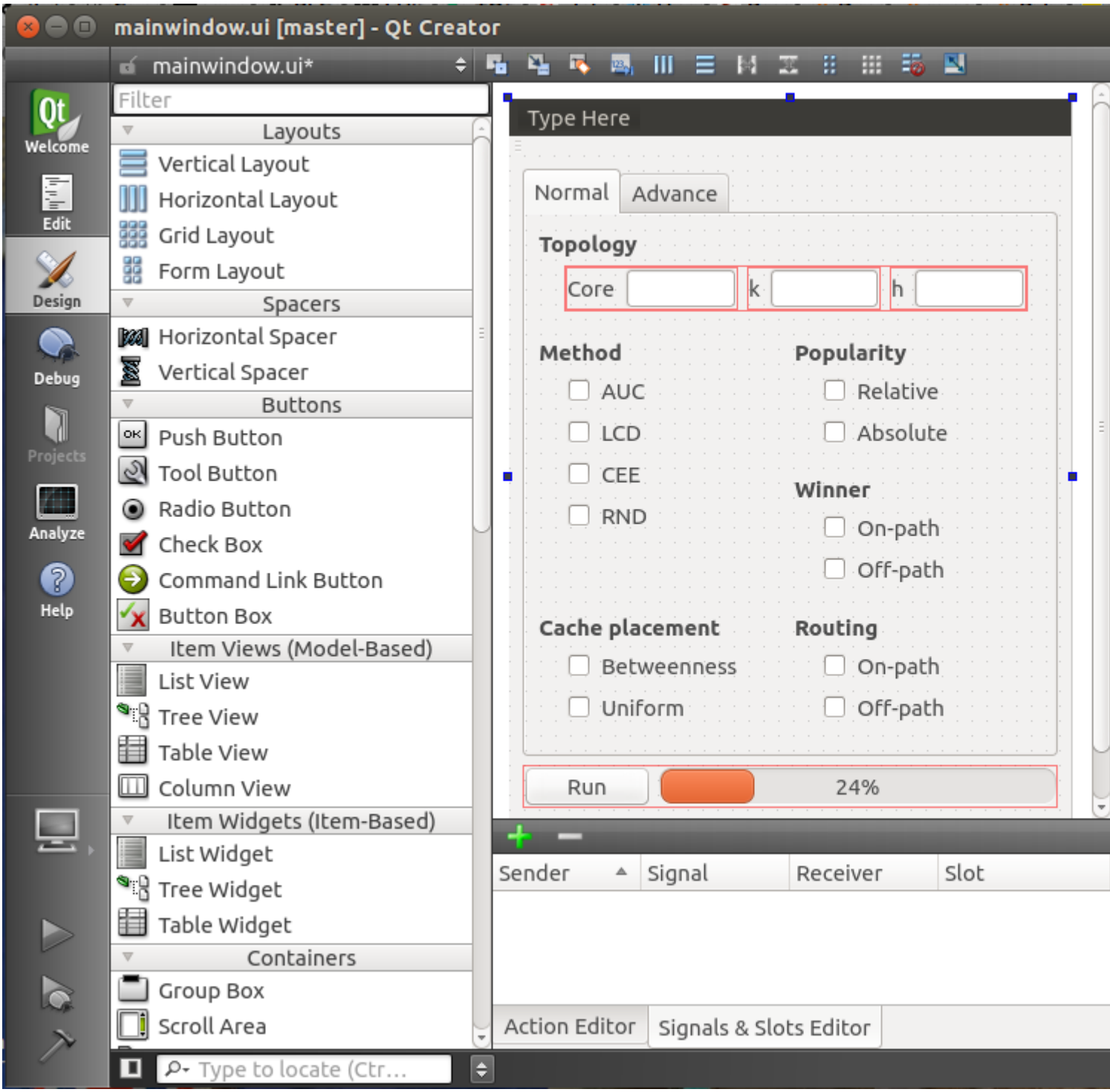
```
import sys
from PyQt5 import QtWidgets

# create instance of QApplication
app = QtWidgets.QApplication(sys.argv)

# create QLabel, without parent it will be shown as window
label = QtWidgets.QLabel('Hello world!')
label.show()

# start the execution loop of the application
sys.exit(app.exec_())
```

Un simple exemple de glisser-déposer

Créez une application graphique simple en 3 étapes faciles.

1. Design

Ouvrez `Qt Creator` , créez un nouveau projet et créez votre projet. Enregistrez votre résultat sous `.ui` fichier `.ui` (ici: `mainwindow.ui`).



2. Générez le fichier .py correspondant

Vous pouvez maintenant créer un fichier .py à partir de votre fichier .ui que vous avez généré à l'étape précédente. Entrez ce qui suit dans votre ligne de commande:

```
$ pyuic4 mainwindow.ui -o GUI.py
```

Si la ligne ci-dessus est exécutée avec succès, un fichier `GUI.py` est créé.

3. Codes Python

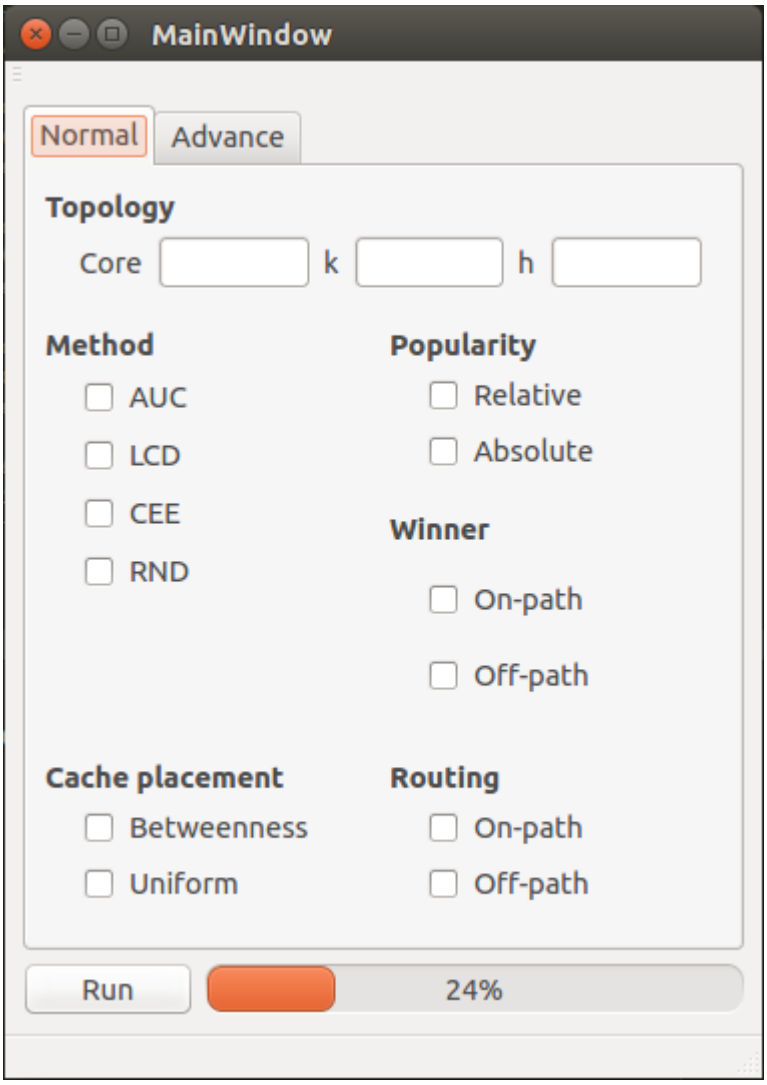
Vous pouvez ajouter votre propre code (par exemple des signaux et des slots) dans le fichier

`GUI.py` , mais il est préférable de les ajouter dans un nouveau fichier. Si vous souhaitez modifier votre interface graphique, le fichier `GUI.py` sera remplacé. C'est pourquoi il est préférable d'utiliser un autre fichier pour ajouter des fonctionnalités dans la plupart des cas.

Appelons le nouveau fichier `main.py`

```
from PyQt4 import QtGui
import sys
import GUI # Your generated .py file


class MyApp(QtGui.QMainWindow, GUI.Ui_MainWindow):
    def __init__(self, parent=None):
        super(ExampleApp, self).__init__(parent)
        self.setupUi(self)

        # Connect a button to a function
        self.btn_run.clicked.connect(self.run)

    def run(self):
        # Write here what happens after the button press
        print("run")


if __name__ == '__main__':
    app = QtGui.QApplication(sys.argv)
    form = ExampleApp()
    form.show()
    app.exec_()
```

Maintenant, vous pouvez exécuter `main.py` et voir votre interface graphique.



Lire Démarrer avec pyqt en ligne: https://riptutorial.com/fr/pyqt/topic/1715/demarrer-avec-pyqt

Chapitre 2: Utiliser des threads avec PyQt

Remarques

Bien que certaines parties du framework Qt soient sûres, la plupart ne le sont pas. La documentation Qt C ++ fournit un bon aperçu des classes réentrantes (peuvent être utilisées pour instancier des objets dans plusieurs threads). Les règles suivantes sont les plus recherchées:

- Vous ne pouvez pas créer ou accéder à un objet graphique Qt en dehors du thread principal (par exemple, tout ce qui sous-classe `QWidget` ou similaire).
- Même si la classe Qt est réentrante, vous ne pouvez pas partager l'accès à un objet Qt entre les threads, sauf si la documentation Qt de cette classe indique explicitement que les instances sont sûres pour les threads.
- Vous pouvez utiliser `QObject.moveToThread()` si vous devez déplacer un objet Qt d'un thread vers un autre (ne s'applique pas aux objets d'interface graphique Qt qui doivent toujours rester dans le thread principal). Mais notez que l'objet ne doit pas avoir de parent.

Selon ce QA de Stack Overflow, il n'est pas recommandé d'utiliser les threads Python si votre thread a l'intention d'interagir avec PyQt de quelque manière que ce soit (même si cette partie du framework Qt est thread-safe).

Examples

Le modèle de travailleur

```
# this method can be anything and anywhere as long as it is accessible for connection
@pyqtSlot()
def run_on_complete():

    pass

# An object containing methods you want to run in a thread
class Worker(QObject):
    complete = pyqtSignal()

    @pyqtSlot()
    def a_method_to_run_in_the_thread(self):
        # your code

        # Emit the complete signal
        self.complete.emit()

# instantiate a QThread
thread = QThread()
# Instantiate the worker object
worker = Worker()
# Relocate the Worker object to the thread
worker.moveToThread(thread)
# Connect the 'started' signal of the QThread to the method you wish to run
thread.started.connect(worker.a_method_to_run_in_the_thread)
```

```
# connect to the 'complete' signal which the code in the Worker object emits at the end of the
method you are running
worker.complete.connect(run_on_complete)
# start the thread (Which will emit the 'started' signal you have previously connected to)
thread.start()
```

Lire Utiliser des threads avec PyQt en ligne: https://riptutorial.com/fr/pyqt/topic/2775/utiliser-des-threads-avec-pyqt

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec pyqt	101, Achayan, Community, Ian, Makoto, mehD, three_pineapples, Trilarion
2	Utiliser des threads avec PyQt	ederag, ekhumoro, three_pineapples