

Python & Qt: Découverte & Utilisation simple

Introduction

Le but de ce mini-tutoriel est de faire découvrir Python et la bibliothèque graphique de Qt version Python : PyQt.

Python est, tout comme PHP, un langage interprété, c'est à dire que le code écrit n'est pas compilé par nos soins.

Qt est une bibliothèque graphique développé par Nokia offrant plusieurs solutions de développement de fenêtrage, interaction avec l'utilisateur, etc...

Ce mini-tutoriel s'adresse à des utilisateurs de Python mais pas Qt.

Sommaire

I - Installation de Python 3.1.2 & PyQt 4

II - Hello World de test

III - Création d'une fenêtre

IV - Création d'un bouton

V - Utiliser les signaux et les slots

I - Installation de Python 3.1.2 & PyQt 4

L'installation de Python n'a rien de compliqué, cette partie est juste là à titre de rappel.

Python sous Windows :

Exécutable : [Python 3.1.2](#)

Mais pour plus de prudence, visitez la page de téléchargement de Python : [Python Download](#)

Python inclue beaucoup de ressources dans sa bibliothèque standard, mais pas Qt :) la bibliothèque graphique de Python est Tkinter.

PyQt sous Windows :

Exécutable binaire : [PyQt 4.7.4-1 pour Python 3.1.x](#)

Attention car Python développe deux versions, la 2.7 & la 3.1.2, il existe donc des installations de PyQt pour les 2.

PyQt intègre QtDesigner, qui permet de dessiner ces propres fenêtres ainsi que d'autres outils.

II - Hello World de test

Le but ici est de tester que Python et PyQt fonctionne correctement.

Celui-ci est tiré de Wikipédia : [Page Qt de Wikipedia](#), celui-ci étant très simple et fonctionne parfaitement, pas besoin d'aller plus loin.

```
from PyQt4 import QtGui, QtCore
import sys

app = QtGui.QApplication(sys.argv)
hello = QtGui.QPushButton("Hello World!", None)
hello.show()
```

```
app.exec_()
```

Ceci devrait ouvrir une fenêtre avec un bouton marqué "Hello World".

III - Création d'une fenêtre

Maintenant que l'on est sûr que notre installation fonctionne correctement, nous allons pouvoir commencer avec Qt.

Python étant un langage objet, nous allons donc créer une classe fenêtre qui sera notre objet à appeler.

```
# -*- coding: utf8 -*-
# Ma première fenêtre

from PyQt4 import QtGui, QtCore
import sys

class fenetre(QtGui.QWidget):
    """Classe permettant de créer une fenêtre"""
    def __init__(self, parent=None):
        """Constructeur permettant d'initialiser la fenêtre à son appel"""
        QtGui.QWidget.__init__(self, parent)

        self.setWindowTitle("Ma première fenêtre")
        self.resize(332, 190)

app = QtGui.QApplication(sys.argv)
f = fenetre()
f.show()
sys.exit(app.exec_())
```

Ici nous avons plusieurs choses à détailler, je ne m'attarderai que sur les aspects de Qt, pas sur le Python.

Dans notre classe fenetre (qui dérive de QtGui.QWidget), nous avons uniquement notre Constructeur (appelé `__init__` en Python).

Dans celui-ci, nous indiquons qu'il faut appeler le constructeur de l'objet QWidget.

Ensuite, deux utilisations simples : changer le titre, et changer la taille de la fenêtre, ceci très simplement.

Et pour finir l'appel de notre objet fenetre().

A noter qu'il faut appeler auparavant l'objet QApplication, qui initialise Qt.

IV - Création d'un bouton

Un bouton dans Qt est un QPushButton, il permet de valider un formulaire comme en HTML par exemple, ou d'activer une séquence.

Voici sa syntaxe :

```
QtGui.QPushButton("Mon Bouton", self)
```

Le premier paramètre permet de définir le texte, le second permet de dire qui est son parent, ici self. Si on remplace ce bouton dans notre première fenêtre le code devient celui-ci :

```
# -*- coding: utf8 -*-
```

```

# Ma première fenêtre

from PyQt4 import QtGui, QtCore
import sys

class fenetre(QtGui.QWidget):
    """Classe permettant de créer une fenêtre"""
    def __init__(self, parent=None):
        """Constructeur permettant d'initialiser la fenêtre à son appel"""
        QtGui.QWidget.__init__(self, parent)

        # La fenêtre
        self.setWindowTitle("Ma première fenêtre")
        self.resize(332, 190)

        # Les boutons
        self.mon_bouton = QtGui.QPushButton("Hey !", self)

app = QtGui.QApplication(sys.argv)
f = fenetre()
f.show()
sys.exit(app.exec_())

```

Dans cette fenêtre vous devriez avoir un bouton en haut à gauche. A noter que les composants de Qt prennent compte du "design" de l'environnement, ainsi sous chaque système d'exploitation, ce bouton aura son apparence propre.

Quand je disais précédemment que ce bouton s'apparentait à l'HTML, Qt permet également de lui associé une feuille de style, comme ceci :

```
self.mon_bouton.setStyleSheet("background-color: green;")
```

Celui-ci deviendra vert, mais il perd donc l'apparence du système d'exploitation. Donc pour ceux qui connaissent les feuilles de style (CSS), il est facile de modifier plusieurs de ses caractéristiques.

```
self.mon_bouton.setStyleSheet("color : white;")
```

Pour voir tout ce que cet objet peut faire (et il peut faire beaucoup de choses !) :

```
print(dir(self.mon_bouton))
```

Mais c'est bien beau d'avoir un bouton mais si on ne peut rien en faire, ce n'est pas très utile ! Qt introduit un fonctionnement, les signaux et les slots, qui vont être détaillé dans la partie suivante.

V - Utiliser les signaux et les slots

Dans Qt, pour donner une utilité à un objet, il existe ce qu'ils sont créé : les signaux et les slots.

Par exemple, un bouton émet un signal lorsqu'on a cliqué sur celui-ci, et le slot est l'action que l'on va dans ce cas-là.

Pour permettre l'utilisation de ce système, il faut utiliser la fonction connect(), comme ceci :

```
self.connect(self.mon_bouton, QtCore.SIGNAL("clicked()"), self.mon_click)
```

Ici, on donne le bouton qui va émettre le signal, le type de signal, (dont je donnerai une liste après), et pour finir l'action à réaliser (le slot). Si nous reprenons l'exemple complet du dessus, cela donne :

```
# -*- coding: utf8 -*-
# Ma première fenêtre

from PyQt4 import QtGui, QtCore
import sys

class fenetre(QtGui.QWidget):
    """Classe permettant de créer une fenêtre"""
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)

        # La fenêtre
        self.setWindowTitle("Ma première fenêtre")
        self.resize(332, 190)

        # Les boutons
        self.mon_bouton = QtGui.QPushButton("Hey !", self)

        # Signaux
        self.connect(self.mon_bouton, QtCore.SIGNAL("clicked()"),
self.click)

    def click(self):
        """Clique sur le bouton"""
        # Ouverture de la page
        self.mon_bouton.setText("Cliqué !")

app = QtGui.QApplication(sys.argv)
f = fenetre()
f.show()
sys.exit(app.exec_())
```

Ici lorsqu'on clique sur le bouton, l'appel du slot allant vers la méthode click est réalisé, et le bouton change de texte.

Quelques signaux disponibles pour un bouton : clicked(), released(), pressed(), toggled(), destroyed(), etc... À vous d'essayer ces signaux :)