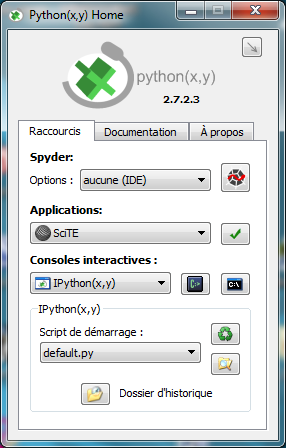
Tutoriel Qt Designer + Python

# Etape 1 : Python(x,y)

Plusieurs solutions s’offrent à nous pour développer une application fenêtrée en python. J’ai choisi la suite Python(x,y) car elle dispose de tous les outils nécessaires regroupés dans une seule application. L’installation des outils se réduit donc à l’installation du logiciel Phython(x,y).



Consoles python

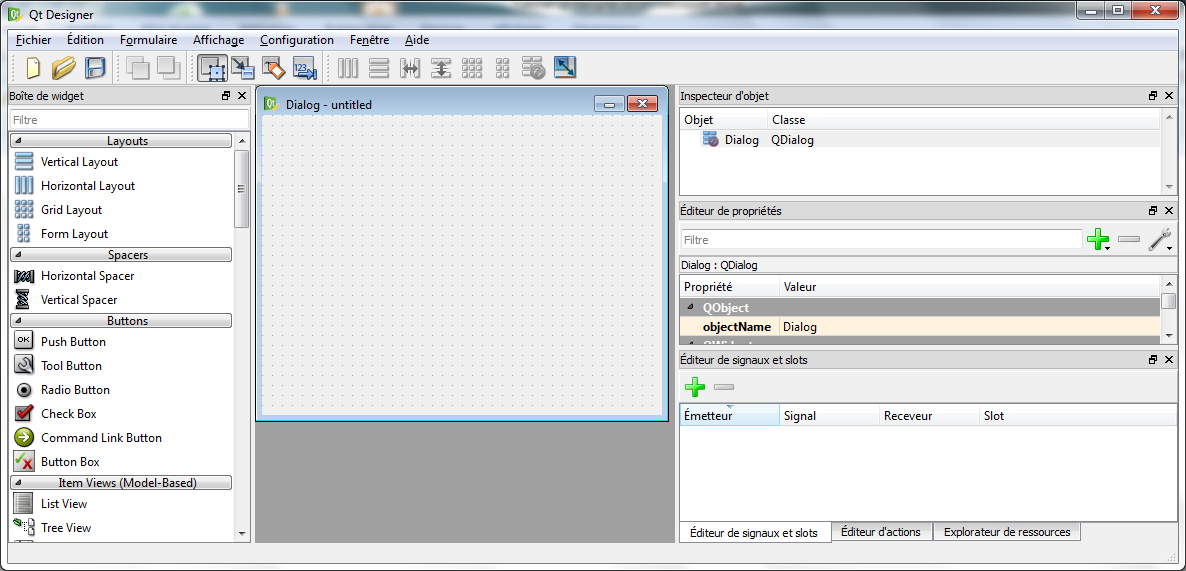
Editeur Python + débogueur + console d’exécution

Editeur Python + débogueur + console d’exécution

# Etape 2 : Interface utilisateur (UI)

Qt Designer permet de créer graphiquement l’interface utilisateur ainsi que la gestion des évènements (signaux et slots).

Dans la fenêtre Home de Python(x,y), choisir Qt Designer dans la liste des applications disponibles et cliquez sur le bouton de validation.



Utilisation de Qt Designer :

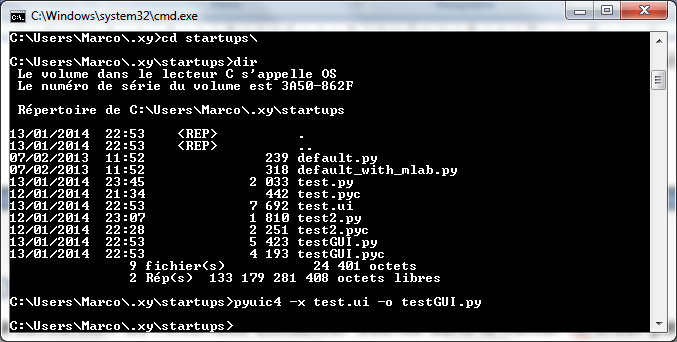
<http://fr.openclassrooms.com/informatique/cours/programmez-avec-le-langage-c/modeliser-ses-fenetres-avec-qt-designer>

Qt Designer créer un fichier .ui qui contient la description de la fenêtre au format xml. Cette description peut être convertie dans différents langages de programmation.

# Etape 2 : Conversion .ui en .py

La conversion de l’interface utilisateur décrite dans le fichier .ui créer par Qt Designer doit être traduite en un script python.

On utilise pour cela la commande : **pyuic4 -x fichier.ui -o fichierGUI.py**



|  |  |
| --- | --- |
| .ui généré par Qt Designer | .py généré par la commande pyuic4 |

# Etape 3 : Application python

On crée une nouvelle application python dans laquelle on importe les librairies suivantes :

|  |
| --- |
| **import** sys  **from** PyQt4.QtGui **import** \*  **import** fichierGUI  # fichierGUI est le fichier .py généré par pyuic4 |

On crée ensuite la classe **MainDialog** correspondant à la fenêtre de l’application :

|  |
| --- |
| **class** **MainDialog**(QDialog,testGUI.Ui\_Dialog):  **def** **\_\_init\_\_**(*self*,parent=None):  super(MainDialog,*self*).\_\_init\_\_(parent)  self.setupUi(*self*) |

Dans cette classe, on décrit aussi les méthodes correspondantes aux slots de connexion aux signaux

|  |  |
| --- | --- |
|  |  |

|  |
| --- |
| **def** **ButtonClicked**(*self*):  *self*.label.setText("Bonjour "+*self*.Edit.text())  **def** **MessageBox**(*self*):  QMessageBox.information(*self*,"Popup","Ceci est un message de test") |

On exécute ensuite l’application :

|  |
| --- |
| app=QApplication(sys.argv)  form=MainDialog()  form.show()  app.exec\_() |