



Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca

Tutorial de desarrollo de aplicaciones con interfaz grafica en Python y Qt (PyQt) - I (5

Per Daniel Rodriguez, [DaniRC](http://www.ibiza-beach.com/) (<http://www.ibiza-beach.com/>)

Creado el 31/08/2006 02:18 modificado el 31/08/2006 02:18

*Esta es la primera parte de una serie de articulos que nos tiene que llevar a crear preciosas aplicaciones graficas con Py
atacaremos las tablas y alguna cosa grafica :-)*

Tutorial de construcción de interfaces gráficas de usuario con F

Partiendo desde cero, llegaremos al infinito y más allá.

Lo primero o como dicen los sajones "Getting started".

En esta ocasión y como ando harto de tanta distro diferente de Linux, voy a centrarme en el uso de software libre en Windows.

Para empezar nos pillamos todo el software que nos va a hacer falta:

- [Python2.4.3](http://python.org)⁽¹⁾ de python.org
- [Qt-win-opensource-4.1.2-mingw](http://trolltech.com)⁽²⁾ de trolltech.com -el mingw se requiere y se instala automaticamente si lo quereis-
- [PyQt4-gpl-4.0beta1-Py2.4-Qt4.1.2.exe](http://riverbankcomputing.com)⁽³⁾ de riverbankcomputing.com
- [PyScripter.exe](http://mmm-experts.com)⁽⁴⁾ de mmm-experts.com

Y ahora, aunque solo sea para vacilar un rato, veamos algunos modulos extra que podemos encontrar para Python y que van a ser l
compañeros de carrera.

- [matplotlib-0.87.2.win32-py2.4.exe](http://matplotlib.sourceforge.net)⁽⁵⁾ de matplotlib.sourceforge.net
- [Numeric-24.2.win32-py2.4.exe](http://numpy.scipy.org)⁽⁶⁾ de numpy.scipy.org
- [numarray-1.5.1.win32-py2.4.exe](http://numpy.scipy.org)⁽⁷⁾ de numpy.scipy.org
- [numpy-0.9.6r1.win32-py2.4.exe](http://numpy.scipy.org)⁽⁸⁾ de numpy.scipy.org

Nota: Segun dice la web de numpy.scipy.org numpy viene a reemplazar a los otros 2 paquetes numarray y numeric. Bueno, pues m
esta allí para que la utilicemos, mientras tanto y como yo usaba estos paquetes antes de que inventaran el numpy, sigo en mis 13 y

Instalación

Una vez que lo teneis todo descargado, lo instalamos en el mismo orden en que lo he ido listando. Se puede usar otro orden, pero b
funcionara seguro, que es de lo que se trata al final.

Documentación

Si algo tiene python es una gran comunidad que lo apoya y una documentación a la altura de los mas grandes. Ademas de la docum
encontrareis en las webs que he ido referenciando antes, tenemos otros manuales de lectura obligada. A saber:

[Inmersion en python](#)⁽⁹⁾ - [Dive into python](#)⁽¹⁰⁾. No hay mejor forma de aprender python que con este fabuloso libro on-line y gratuito
avanzada asi que si sabeis ingles aprovechadlo, sino, la version española tambien os sacara del paso.

Sobre PyQt, lo cierto es que todo lo que se lo he aprendido a base de perder horas delante de la pantalla y sobre todo con el libro of
haber localizado en la web de riverbankcomputing si sois algo listos ;-)



[GUI Programming with PyQt](#)⁽¹¹⁾ solo está en inglés, pero no tiene desperdicio. Si el inglés es un problema, tranquilos que para eso

Por lo demas, ya estais tardando en agregar a favoritos estos enlaces de vital importancia:

- [API de Qt de Trolltech](#)⁽¹²⁾, la misma que usaremos desde Python.
 - [Documentacion de python en ingles](#)⁽¹³⁾ y [Documentacion de python en castellano](#)⁽¹⁴⁾ -como de costumbre en las traducciones mejor que nada-
-

Empezamos con la 1ª aplicacion

Lo que haremos aqui no seran complicados proyectos MDI, sino aplicaciones rapidas de andar por casa para que los trabajos que p... Asi que voy al grano:

Primero arrancar el QtDesigner que esta o en el escritorio o en el menu de inicio seccion programas, qt y esas cosas. Sinceramente encontrar un icono mejor dejad este tutorial chicos y pillaos uno de windows.

El qtdesigner es un juguetito que no tiene mayores complicaciones, peor voy a hacer un paso a paso para que no os quejeis:

Primero haremos una **MAIN WINDOW** o ventana principal, a la que le haremos un menu desde el que llamaremos a los dialogos logica de la aplicacion.

Lo siguiente sera hacer el menu. Con que tenga una opción nos bastará, luego es siempre lo mismo.



Veamos lo que he hecho. He escrito Menu1 en la barra de menu. Al Menu1 le he añadido la Opcion1 y listos. Luego He ido al panel de propiedades. Con la ventana principal seleccionada he cambiado el windowTitle por el de "ventana principal" y he guardado la ventana principal con la extension .ui

Voy a llamarla por ejemplo: principal.ui

Y ya esta, ya tenemos media aplicacion en marcha, con su menu y todo! Vamos a verlo.

Compilar un formulario .ui a su version .py

Para la ocasión tenemos que usar el modo consola de windows, el comando como lo querreis llamar.

Abriremos una consola de sistema (cmd) que nos de acceso al fichero .ui que acabamos de crear. Y lanzaremos el siguiente comando

pyuic4 -o principal.py -x principal.ui

Esto se lee como:

python user interface compiler para Qt4 genera un output de nombre principal.py que es ejecutable y que se basa en lo que dice el fichero .ui o si lo preferis, lo que hace es convertir un fichero xml de definicion de formularios en un formulario usable desde Python.

PROBLEMAS: El problema mas comun en este momento es que no os reconozcan la orden pyuic4. Eso es debido a un problema con el path juntitos:

Boton derecho sobre miPC -> propiedades -> Opciones Avanzadas -> Variable de Entorno

en la parte de arriba, donde dice Variables de entorno del usuario añadís la variable **PATH** si no esta creada. Si esta creada la modificáis añadiendo las siguientes entradas:

C:\Qt\4.1.2\bin;C:\Python24;C:\Python24\bin;C:\Python24\lib;C:\Python24\lib\site-packages;C:\Python24\dll

Y ahora si guardais esos cambios y cerrais la consola de sistema y la volveis a abrir el problema tiene que haberse esfumado. Si dice que no funciona incluso reiniciar, pero no debería ser necesario.



¿Qué es Principal.py?

Si todo ha ido bien, tenemos ahora un principal.py junto a nuestro principal.ui

Veamos que es cada cosa:

Principal.ui	Principal.py
<pre><ui version="4.0" > <author></author> <comment></comment> <exportmacro></exportmacro> <class>MainWindow</class> <widget class="QMainWindow" name="MainWindow" > <property name="geometry" > <rect> <x>0</x> <y>0</y> <width>800</width> <height>600</height> </rect> </property> <property name="windowTitle" > <string>Ventana Principal</string> </property> <widget class="QWidget" name="centralwidget" /> <widget class="QMenuBar" name="menubar" > <property name="geometry" > <rect> <x>0</x> <y>0</y> <width>800</width> <height>21</height> </rect> </property> <widget class="QMenu" name="menuMenu1" > <property name="title" > <string>Menu 1</string> </property> <addaction name="actionOpcion1" /> </widget> <addaction name="menuMenu1" /> </widget> <widget class="QStatusBar" name="statusbar" > <property name="geometry" > <rect> <x>0</x> <y>581</y> <width>800</width> <height>19</height> </rect> </property></pre>	<pre># -*- coding: utf-8 -*- # Form implementation generated from reading ui file 'unknown' # # Created: Thu Aug 31 00:42:11 2006 # by: PyQt4 UI code generator 4.0beta1 # # WARNING! All changes made in this file will be lost! import sys from PyQt4 import QtCore, QtGui class Ui_MainWindow(object): def setupUi(self, MainWindow): MainWindow.setObjectName("MainWindow") MainWindow.resize(QtCore.QSize(QtCore.QRect(0,0,800,600).size()).expandedTo(MainWindow.size())) self.centralwidget = QtGui.QWidget(MainWindow) self.centralwidget.setObjectName("centralwidget") MainWindow.setCentralWidget(self.centralwidget) self.menubar = QtGui.QMenuBar(MainWindow) self.menubar.setGeometry(QtCore.QRect(0,0,800,21)) self.menubar.setObjectName("menubar") self.menuMenu1 = QtGui.QMenu(self.menubar) self.menuMenu1.setObjectName("menuMenu1") MainWindow.setMenuBar(self.menubar) self.statusbar = QtGui.QStatusBar(MainWindow) self.statusbar.setGeometry(QtCore.QRect(0,581,800,19)) self.statusbar.setObjectName("statusbar") MainWindow.setStatusBar(self.statusbar) self.actionOpcion1 = QtGui.QAction(MainWindow) self.actionOpcion1.setObjectName("actionOpcion1") self.menuMenu1.addAction(self.actionOpcion1) self.menubar.addAction(self.menuMenu1.menuAction()) self.retranslateUi(MainWindow) QtCore.QMetaObject.connectSlotsByName(MainWindow) def tr(self, string): return QtGui.QApplication.translate("MainWindow", string, None, QtGui.QApplication.UnicodeUTF8) def retranslateUi(self, MainWindow): MainWindow.setWindowTitle(self.tr("Ventana Principal")) self.menuMenu1.setTitle(self.tr("Menu 1")) self.actionOpcion1.setText(self.tr("Opcion 1"))</pre>



<pre> </widget> <action name="actionOpcion1" > <property name="text" > <string>Opcion1</string> </property> </action> </widget> <pixmapfunction></pixmapfunction> <resources/> <connections/> </ui> </pre>	<pre> if __name__ == "__main__": app = QtGui.QApplication(sys.argv) MainWindow = QtGui.QMainWindow() ui = Ui_MainWindow() ui.setupUi(MainWindow) MainWindow.show() sys.exit(app.exec_()) </pre>
--	---

Como podeis ver, principal.ui era un fichero de texto etiquetado en xml. Sin embargo, **principal.py ya es codigo python y puede ser ejecutado con un simple doble click.**

IMPORTANTISIMO: Los formularios los hacemos en QtDesigner, que genera un fichero.ui que traducimos a python con el traductor. Sin embargo, tanto, aquí no hemos codificado aun ni una sola linea de codigo a mano y aun mas importante. Cualquier cosa que codifiquemos a mano perderemos si tenemos que volver a generarlos!! Es decir, si quereis cambiar algo hacedlo desde la interfaz de QtDesigner, no desde el codigo. **hacer una nueva traduccion del .ui a .py**

De momento, nuestra aplicacion no vale para gran cosa. Hagamos ahora un dialogo que sea llamado desde nuestra aplicacion.

Como llamar a un dialogo desde nuestra aplicacion principal.

Lo primero hacer el dialogo desde Qt Designer, pero no le digais Dialog decidle que quereis hacer un **WIDGET** no es obligatorio poner los botones que pone el diseñador automaticamente cada vez.

Despues de eso, poned visible el panel del editor de propiedades (menu tools, property editor)

Ahora demosle un nombre normal al formulario, yo lo llamo: frmAlgo

Le pongo un titulo en la propiedad windowTitle: Mi Dialogo

Arrastro elementos de formulario desde el panel de elementos de diseño hasta el formulario. En concreto voy a pedir 2 numeros al usuario y esos numeros segun a que boton se le pulse. El resultado saldra en el 3er cuadro de texto.



3 etiquetas (Label) para poner el nombre de los campos visible.
3 componentes QLineEdit para insertar texto en ellos.
2 botones QPushButton para hacer los botones de sumar y restar.

Las etiquetas no las voy a usar, así que simplemente les cambio el texto -boton derecho, change text-
Los cuadros de texto, los llamo txtOperando1, txtOperando2, txtResultado para poder identificarlos con facilidad en el código.
Los botones los llamo btnSumar y btnRestar, para poder identificarlos con facilidad en el código.

No uso los layouts, ni los spacers por simplicidad, voy al grano a que useis esto como un VB5.0 pero mas potente.

Os debería quedar algo como eso de ahí arriba. Ahora lo guardamos con el mismo nombre que el que pusimos al formulario. Si el fichero es llamara frmAlgo.ui y lo compilaremos para convertirlo en -habeis adivinado!- frmAlgo.py

El comando sería tal que así:

```
pyuic4 -o frmAlgo.py -x frmAlgo.ui
```

Con doble click sobre el fichero .py deberíais estar viendo esto:

No tiene ninguna funcionalidad, pero es un principio. El código que habeis autogenerado tiene esta pinta:

```
# -*- coding: utf-8 -*-  
  
# Form implementation generated from reading ui file 'unknown'  
#  
# Created: Thu Aug 31 01:08:52 2006  
# by: PyQt4 UI code generator 4.0beta1  
#  
# WARNING! All changes made in this file will be lost!
```



```

import sys
from PyQt4 import QtCore, QtGui

class Ui_frmAlgo(object):
    def setupUi(self, frmAlgo):
        frmAlgo.setObjectName("frmAlgo")
        frmAlgo.resize(QtCore.QSize(QtCore.QRect(0,0,335,138).size()).expandedTo(frmAlgo.minimumSizeHint()))

        self.label_2 = QtGui.QLabel(frmAlgo)
        self.label_2.setGeometry(QtCore.QRect(10,60,61,22))
        self.label_2.setObjectName("label_2")

        self.label_3 = QtGui.QLabel(frmAlgo)
        self.label_3.setGeometry(QtCore.QRect(10,100,56,22))
        self.label_3.setObjectName("label_3")

        self.label = QtGui.QLabel(frmAlgo)
        self.label.setGeometry(QtCore.QRect(10,20,61,22))
        self.label.setObjectName("label")

        self.btnSumar = QtGui.QPushButton(frmAlgo)
        self.btnSumar.setGeometry(QtCore.QRect(240,20,75,23))
        self.btnSumar.setObjectName("btnSumar")

        self.btnRestar = QtGui.QPushButton(frmAlgo)
        self.btnRestar.setGeometry(QtCore.QRect(240,60,75,23))
        self.btnRestar.setObjectName("btnRestar")

        self.txtOperando1 = QtGui.QLineEdit(frmAlgo)
        self.txtOperando1.setGeometry(QtCore.QRect(70,20,113,20))
        self.txtOperando1.setObjectName("txtOperando1")

        self.txtOperando2 = QtGui.QLineEdit(frmAlgo)
        self.txtOperando2.setGeometry(QtCore.QRect(70,60,113,20))
        self.txtOperando2.setObjectName("txtOperando2")

        self.txtResultado = QtGui.QLineEdit(frmAlgo)
        self.txtResultado.setGeometry(QtCore.QRect(70,100,113,20))
        self.txtResultado.setObjectName("txtResultado")

        self.retranslateUi(frmAlgo)
        QtCore.QMetaObject.connectSlotsByName(frmAlgo)

    def tr(self, string):
        return QtGui.QApplication.translate("frmAlgo", string, None, QtGui.QApplication.UnicodeUTF8)

    def retranslateUi(self, frmAlgo):
        frmAlgo.setWindowTitle(self.tr("Mi Dialogo"))
        self.label_2.setText(self.tr("<html><head><meta name='qrichtext' content='1' /></head><body style=' white-space: pre-wrap; font-size:8.25pt; font-weight:400; font-style:normal; text-decoration:none;'><p style=' margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; text-align:center;'><span style=' font-size:8pt;'>Operando2</span></p></body></html>"))
        self.label_3.setText(self.tr("<html><head><meta name='qrichtext' content='1' /></head><body style=' white-space: pre-wrap; font-size:8.25pt; font-weight:400; font-style:normal; text-decoration:none;'><p style=' margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; text-align:center;'><span style=' font-size:8pt;'>Resultado</span></p></body></html>"))
        self.label.setText(self.tr("<html><head><meta name='qrichtext' content='1' /></head><body style=' white-space: pre-wrap; font-size:8.25pt; font-weight:400; font-style:normal; text-decoration:none;'><p style=' margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; text-align:center;'><span style=' font-size:8pt;'>Operando 1</span></p></body></html>"))
        self.btnSumar.setText(self.tr("Sumar"))
        self.btnRestar.setText(self.tr("Restar"))

```



```

if __name__ == "__main__":
    app = QtGui.QApplication(sys.argv)
    frmAlgo = QtGui.QWidget()
    ui = Ui_frmAlgo()
    ui.setupUi(frmAlgo)
    frmAlgo.show()
    sys.exit(app.exec_())

```

De nuevo se nos informa de que este formulario ha sido autogenerado y seria un error sobreescribirlo porque en cuanto cambiemos perder. La solución mas inteligente es extender esta clase y poner toda la parte de "funcionalidad" en la clase que extienda a esta. Y "manual" de programación. Lo bueno es que como es algo muy mecanico pronto lo haremos con soltura y facilidad.

Estructura de directorios y paquetes de la aplicación

Lo primero es tener un nombre bonito para el proyecto. Es corriente usar la palabra py en el nombre, como pynumeric, pyarray, nu voy a llamarlo pydanirc y voy a crear una carpeta pydanirc en la que meter lo que vaya haciendo relativo al proyecto.

Luego sugiero por experiencia -realice sendos trabajos semi-profesionales con pyqt en el 2002 pyecole y pybiblio- la siguiente estructura de directorio principal:

- forms ui - donde metemos todos los .ui
- forms - donde metemos los forms que hemos generado con pyuic
- dialogs - donde metemos las clases que extienden al formulario y lo dotan de funcionalidad
- reports - para los informes
- graphics - para las graficas

La pantalla principal, debería estar en la raíz del directorio principal.

Por su parte todo debería estar dentro de **C:\Python24\Lib\Site-packages** que es donde se meten los modulos de python hechos por nosotros.

Por ultimo y no menos importante, para que python sepa que lo que estamos haciendo son modulos tenemos que poner en cada una un nombre **__init__.py** (dos guiones bajos a cada lado)

Ese fichero **__init__.py** puede servir para inicializaciones pero en nuestro caso lo vamos a dejar vacio. Yo siempre le pongo dentro

#No lo borren aunque este vacio.

IMPORTANTE: Meted uno de esos **__init__.py** **en cada carpeta** no sólo en la principal! en dialogs y en forms también, en 'form' habeis puesto tampoco pasa nada.

Arrancamos el PyScripter para dar funcionalidad al formulario!



Este es el código en concreto:

```
import sys
from PyQt4 import QtCore, QtGui
from pydanirc.forms.frmAlgo import Ui_frmAlgo

class dlgAlgo(QtGui.QDialog):

    def __init__(self):

        QtGui.QDialog.__init__(self)

        #Inicializar el formulario
        self.ui = Ui_frmAlgo()
        self.ui.setupUi(self)

        # Conectar los botones con funciones del dialogo.
        self.connect(self.ui.btnSumar, QtCore.SIGNAL("clicked()"),self.suma)
        self.connect(self.ui.btnRestar, QtCore.SIGNAL("clicked()"),self.resta)

    def suma(self):

        a = int(self.ui.txtOperando1.text())
        b = int(self.ui.txtOperando2.text())
        c = a + b
        self.ui.txtResultado.setText(str(c))

    def resta(self):

        a = int(self.ui.txtOperando1.text())
        b = int(self.ui.txtOperando2.text())
        c = a - b
        self.ui.txtResultado.setText(str(c))
```

Por supuesto, si el formulario era frmAlgo, esto es el dlgAlgo guardado con el nombre de dlgAlgo.py en la carpeta de dialogos .

Bueno, más fácil no se hacerlo, y python es tan limpio y bonito que se explica solo.

Importamos el formulario frmAlgo del paquete danirc de la carpeta formularios.

Lo inicializamos con el setupUi y luego hacemos que el botón sumar responda al evento click con una llamada a la función suma y



`self.ui.NOMBRECOMPONENTE.METODO` es la forma normal de acceder a un componente visual para recoger o para setear un `asi de simple` parece y `asi de simple` es.

Ahora bien ... falta llamar al dialogo desde algun lado, por ejemplo desde el menu de la ventana principal!

Llamando dialogos desde los menus de la ventana principal

Tenemos que romper una regla de oro importante que era, NO TOCAR los autogenerados. Bueno, pues en esta ocasion vamos a to concreto le vamos a añadir un par de lineas.

Para empezar arriba del todo hacemos accesible el dialogo para la pantalla principal

```
from pydanirc.dialogs.dlgAlgo import dlgAlgo
```

Algo mas abajo, como ultima linea del `setupUi` de la ventana principal conectamos la opcion del menu con una funcion que nos lan necesario.

```
self.connect(self.actionOpcion1, QtCore.SIGNAL("triggered()"), self.abreDlgAlgo)
```

para acabar creamos la funcion (metodo para hablar con mas propiedad) que lanzara el dialogo

```
def abreDlgAlgo(self):
```

```
    d = dlgAlgo()
    d.exec_()
```

y ahora una cosa "rara". Vamos a cambiarle el objeto del que hereda la clase de la ventana principal

donde dice:

```
class Ui_MainWindow(object):
```

ponemos

```
class Ui_MainWindow(QtGui.QMainWindow):
```

y ahora con toda calma probamos nuestra fantastica aplicación con un doble click sobre el fichero que dice `principal.py` :-)

No intenteis arrancar esto desde `pyscripter` porque suele colgarse ya que se lanza como un hilo hijo del `scripter` ... y en fin ... una la y mejor lo haceis con un doble click sobre el iconito de `principal.py` y listos.



NOTA: Si, la ventana negra de la consola de sistema de quedara ahi detras para siempre. En general si no la usamos para mensajes pero nunca lo he intentado, ni me molesta para nada asi que tampoco perdere mi tiempo en mirarlo, si alguien lo descubre que lo diga agradeceremos -que no todo va a ser recibir! cony! alguna vez tambien teneis que dar ;-)-

Se me olvidaba! El codigo de principal.py ya modificado por si alguno tuvo dificultades.

```
from pydanirc.dialogs.dlgAlgo import dlgAlgo
```

```
import sys
from PyQt4 import QtCore, QtGui
```

```
class Ui_MainWindow(QtGui.QMainWindow):
```

```
    def setupUi(self, MainWindow):
```

```
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(QtCore.QSize(QtCore.QRect(0,0,800,600).size()).expandedTo(MainWindow.minimumSize()))
```

```
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        MainWindow.setCentralWidget(self.centralwidget)
```

```
        self.menubar = QtGui.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0,0,800,21))
        self.menubar.setObjectName("menubar")
```

```
        self.menuMenu1 = QtGui.QMenu(self.menubar)
        self.menuMenu1.setObjectName("menuMenu1")
        MainWindow.setMenuBar(self.menubar)
```

```
        self.statusbar = QtGui.QStatusBar(MainWindow)
        self.statusbar.setGeometry(QtCore.QRect(0,581,800,19))
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
```



```
self.actionOpcion1 = QtGui.QAction(MainWindow)
self.actionOpcion1.setObjectName("actionOpcion1")
self.menuMenu1.addAction(self.actionOpcion1)
self.menuubar.addAction(self.menuMenu1.menuAction())

self.retranslateUi(MainWindow)
QtCore.QMetaObject.connectSlotsByName(MainWindow)

self.connect(self.actionOpcion1, QtCore.SIGNAL("triggered()"), self.abreDlgAlgo)

def tr(self, string):

    return QtGui.QApplication.translate("MainWindow", string, None, QtGui.QApplication.UnicodeUTF8)

def retranslateUi(self, MainWindow):

    MainWindow.setWindowTitle(self.tr("Ventana Principal"))
    self.menuMenu1.setTitle(self.tr("Menu1"))
    self.actionOpcion1.setText(self.tr("Opcion1"))

def abreDlgAlgo(self):

    d = dlgAlgo()
    d.exec_()

if __name__ == "__main__":

    app = QtGui.QApplication(sys.argv)
    MainWindow = QtGui.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

Lista de enlaces de este artículo:

1. <http://www.python.org/ftp/python/2.4.3/python-2.4.3.msi>
2. <http://ftp.iasi.roedu.net/mirrors/ftp.trolltech.com/qt/source/qt-win-opensource->
3. <http://www.riverbankcomputing.com/Downloads/PyQt4/GPL/PyQt-gpl-4.0.1-Py2.4-Qt4.1>
4. <http://mmm-experts.com/Downloads.aspx?ProductId=4>
5. http://sourceforge.net/project/showfiles.php?group_id=80706&package_id=82474&rel
6. http://sourceforge.net/project/showfiles.php?group_id=1369&package_id=1351
7. http://sourceforge.net/project/showfiles.php?group_id=1369&package_id=32367
8. <http://prdownloads.sourceforge.net/numpy/numpy-1.0b4.win32-py2.4.exe?download>
9. <http://es.diveintopython.org/>
10. <http://www.diveintopython.org/>
11. <http://www.commandprompt.com/community/pyqt/>
12. <http://doc.trolltech.com/4.2/index.html>
13. <http://www.python.org/doc/>
14. <http://pyspanishdoc.sourceforge.net/>

E-mail del autor: danircJUBILANDOSEbulma.net

Podrás encontrar este artículo e información adicional en: <http://bulma.net/body.phtml?nIdNoticia=2336>