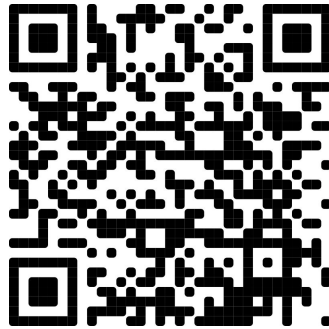




# PyQT5 para GUI en programas escritorio

Por MC. René Solís R.



*Que es PyQT ?*

<https://en.wikipedia.org/wiki/PyQt>

Guia en MAC y Linux

[http://brew.sh/index\\_es.html](http://brew.sh/index_es.html)

<https://www.metachris.com/2016/03/how-to-install-qt56-pyqt5-virtualenv-python3/>

<http://waleedkhan.name/blog/pyqt-designer/#a-brief-guide-to-installi>

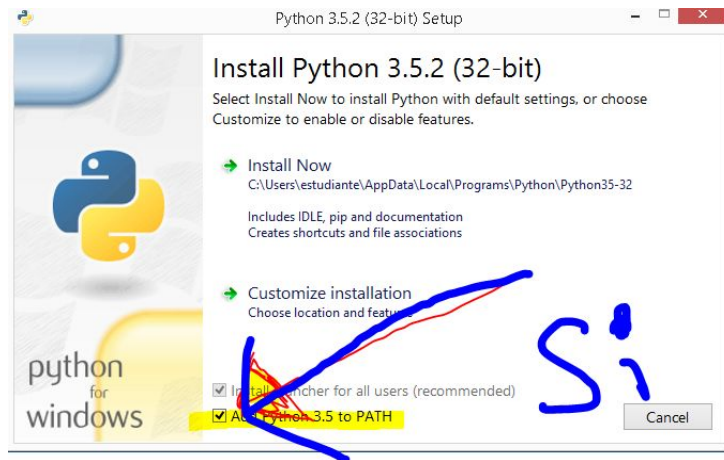
Par este curso estan todo localizado en:

<http://tinyurl.com/ibopython>

(Directorio Utiliza)

# Orden de instalación de los programas

1.- Python 3.x debe dar click a “Incluir PATH” es muy importante seleccionarlo.  
python-3.5.2.exe



*Figura 1. Es importante debemos de activar la opción de PATH (camino de ejecución) para cuando estemos en DOS Shell invocando nuestros programas Windows localice lo necesario para ejecutar correctamente.*

2. Instalar de manera normal y completo PyQt5-5.6-gpl-Py3.5-Qt5.6.0-x32-2.exe
3. Descargar e instalar [notepad-plus-plus.org](http://notepad-plus-plus.org) es openSource solo Windows.
- 4.- Diseñar su pantalla vía QT Designer (icono verde con lápiz) y guardarlo ordenadamente en su folder de prácticas. Tendrá extensión y nombre ejemplo: “holamundo.UI”
- 5.- Para ejecutar la conversión entraremos a tecla Inicio (icono windows) y tecleamos CMD (presionar tecla Enter)
- 6.- Saltamos usando comando en el cuadro negro.

```
cd Desktop (o lugar donde guardó el archivo)  
cd practicas  
pyuic5 -x holamundo.ui -o holamundo.py  
python holamundo.py
```

# Instalación en macOS Sierra

Requerimos una media hora aprox. con un internet estable recomendable minimo 10 mb

1. Vamos y invocamos el APP de nombre TERMINAL y pegamos el comando de shell que esta en el sitio de [http://brew.sh/index\\_es.html](http://brew.sh/index_es.html)
- 2.

```
Last login: Wed Oct 26 09:36:41 on console
[rene-MacBook-Pro:~ rene$ /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
==> This script will install:
/usr/local/bin/brew
/usr/local/share/doc/homebrew
/usr/local/share/man/man1/brew.1
/usr/local/share/zsh/site-functions/_brew
/usr/local/etc/bash_completion.d/brew
/usr/local/Homebrew
==> The following new directories will be created:
/usr/local/Cellar
/usr/local/Homebrew
/usr/local/Frameworks
/usr/local/bin
/usr/local/etc
/usr/local/include
/usr/local/lib
/usr/local/opt
/usr/local/sbin
/usr/local/share
/usr/local/share/zsh
/usr/local/share/zsh/site-functions
/usr/local/var

Press RETURN to continue or any other key to abort
==> /usr/bin/sudo /bin/mkdir -p /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/bin /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
Password:
==> /usr/bin/sudo /bin/chmod g+rxw /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/bin /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /bin/chmod 755 /usr/local/share/zsh /usr/local/share/zsh/site-functions
==> /usr/bin/sudo /usr/sbin/chown rene /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/bin /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /usr/sbin/chgrp admin /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/bin /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /bin/chmod g+rxw /Users/rene/Library/Caches/Homebrew
==> /usr/bin/sudo /usr/sbin/chown rene /Users/rene/Library/Caches/Homebrew
==> Searching online for the Command Line Tools
==> /usr/bin/sudo /usr/bin/touch /tmp/.com.apple.dt.CommandLineTools.installondemand.in-progress
==> Installing Command Line Tools (macOS Sierra version 10.12) for Xcode-8.0

==> /usr/bin/sudo /usr/sbin/softwareupdate -i Command\ Line\ Tools\ (macOS\ Sierra\ version\ 10.12)\ for\ Xcode-8.0
Software Update Tool
Copyright 2002-2015 Apple Inc.

Downloading Command Line Tools (macOS Sierra version 10.12) for Xcode
Downloaded Command Line Tools (macOS Sierra version 10.12) for Xcode
Installing Command Line Tools (macOS Sierra version 10.12) for Xcode
Done with Command Line Tools (macOS Sierra version 10.12) for Xcode
Done.
==> /usr/bin/sudo /bin/rm -f /tmp/.com.apple.dt.CommandLineTools.installondemand.in-progress
==> /usr/bin/sudo /usr/bin/xcode-select --switch /Library/Developer/CommandLineTools
==> Downloading and installing Homebrew...
remote: Counting objects: 3779, done.
remote: Compressing objects: 100% (2559/2559), done.
```

```

==> /usr/bin/sudo /bin/rm -f /tmp/.com.apple.dt.CommandLineTools.installondemand.in-progress
==> /usr/bin/sudo /usr/bin/xcode-select --switch /Library/Developer/CommandLineTools
==> Downloading and installing Homebrew...
remote: Counting objects: 3779, done.
remote: Compressing objects: 100% (2559/2559), done.
remote: Total 3779 (delta 1877), reused 2429 (delta 1080), pack-reused 0
Receiving objects: 100% (3779/3779), 2.26 MiB | 110.00 KiB/s, done.
Resolving deltas: 100% (1877/1877), done.
From https://github.com/Homebrew/brew
* [new branch]      master      -> origin/master
* [new tag]          0.1         -> 0.1
* [new tag]          0.2         -> 0.2
* [new tag]          0.3         -> 0.3
* [new tag]          0.4         -> 0.4
* [new tag]          0.5         -> 0.5
* [new tag]          0.6         -> 0.6
* [new tag]          0.7         -> 0.7
* [new tag]          0.7.1       -> 0.7.1
* [new tag]          0.8         -> 0.8
* [new tag]          0.8.1       -> 0.8.1
* [new tag]          0.9         -> 0.9
* [new tag]          0.9.1       -> 0.9.1
* [new tag]          0.9.2       -> 0.9.2
* [new tag]          0.9.3       -> 0.9.3
* [new tag]          0.9.4       -> 0.9.4
* [new tag]          0.9.5       -> 0.9.5
* [new tag]          0.9.8       -> 0.9.8
* [new tag]          0.9.9       -> 0.9.9
* [new tag]          1.0.0       -> 1.0.0
* [new tag]          1.0.1       -> 1.0.1
* [new tag]          1.0.2       -> 1.0.2
* [new tag]          1.0.3       -> 1.0.3
* [new tag]          1.0.4       -> 1.0.4
* [new tag]          1.0.5       -> 1.0.5
* [new tag]          1.0.6       -> 1.0.6
* [new tag]          1.0.7       -> 1.0.7
* [new tag]          1.0.8       -> 1.0.8
HEAD is now at 84d1661 Merge pull request #1365 from MikeMcQuaid/audit-stable-url-beta
==> Homebrew has enabled anonymous aggregate user behaviour analytics
Read the analytics documentation (and how to opt-out) here:
  https://git.io/brew-analytics
==> Tapping homebrew/core
Cloning into '/usr/local/Homebrew/Library/Taps/homebrew/homebrew-core'...
remote: Counting objects: 3739, done.
remote: Compressing objects: 100% (3629/3629), done.
remote: Total 3739 (delta 12), reused 298 (delta 1), pack-reused 0
Receiving objects: 100% (3739/3739), 2.99 MiB | 376.00 KiB/s, done.
Resolving deltas: 100% (12/12), done.
Checking connectivity... done.
Tapped 3618 formulae (3,766 files, 9.3M)
Already up-to-date.
==> Installation successful!
==> Next steps
Run `brew help` to get started
Further documentation: https://git.io/brew-docs

```

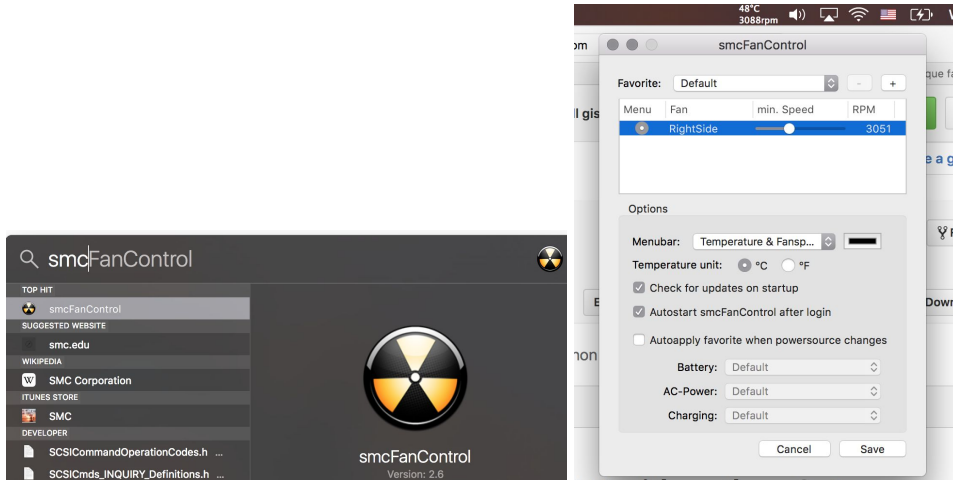
Probaremos si funciona correctamente o falta algo, para eso instalaremos una herramienta famosa para controlar los abanicos de la MAC.

## \$ brew cask install smcfancontrol

```

[renes-MacBook-Pro:~ rene$ sudo su
sh-3.2# brew cask install smcfancontrol
==> Satisfying dependencies
complete
==> Downloading https://www.eidac.de/smcfancontrol/smcfancontrol_2_6.zip
##### 100.0%
==> Verifying checksum for Cask smcfancontrol
==> Moving App 'smcFanControl.app' to '/Applications/smcFanControl.app'
🍺 smcfancontrol was successfully installed!
sh-3.2#

```



*Esto es para probar que BREW está listo para llamar librerías de código opensource de MAC y Linux con extensiones para desarrollo.*

*Dentro de Terminar invocamos la instalación de el URL <https://www.metachris.com/2016/03/how-to-install-qt56-pyqt5-virtualenv-python3/> nada mas que hay unos errores y se mejora en este tutorial.*

### *# Continuamos en nuestra terminal*

```

$ brew install python3
$ xcode-select --install # si instaló SMCfanControl dirá que ya
está listo.

```

### *# Crear directorio de trabajo*

```

$ cd
$ sudo mkdir -p ~/.venv

```

### *# Crear el virtual environment*

```

$ sudo python3 -m venv ~/.venv/qtproject

```

### *# Activar el virtual environment*

```

$ . ~/.venv/qtproject/bin/activate

```

# Todo bien?

\$ which python3

*/Users/rene/.venv/qtproject/bin/python3 # mas o menos asi importante recordarlo mas adelante. Aqui ya invertimos 20 minutos.*

# *Instalar Soporte de python de nombre SIP y pyQT5*

\$ sudo pip install --upgrade pip

\$ sudo pip3 install sip

\$ sudo pip3 install PyQt5

\$ brew cask install sublime-text # *Nuestro editor muy popular Sublime-Text.*

# Ahora importarlo para que sea reconocido por python3

\$ which python

/Users/rene/.venv/qtproject/bin/python # copiamos la respuesta y a pegamos en la siguiente linea...

\$ /Users/rene/.venv/qtproject/bin/python -c "import PyQt5"

*Nuestra primera prueba recuerda que este activado virtualEnv*

```
$ . ~/.venv/qtproject/bin/activate
```

*Salvarlo como hola.py puedes utilizar cualquier editor, recomendable es Sublime ya esta instalado, abajo esta la corrida y como se ve el editor (y donde guardarlo)*

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-

"""
ZetCode PyQt5 tutorial

In this example, we create a simple
window in PyQt5.

author: Jan Bodnar
website: zetcode.com
last edited: January 2015
"""

import sys
from PyQt5.QtWidgets import QApplication, QWidget

if __name__ == '__main__':

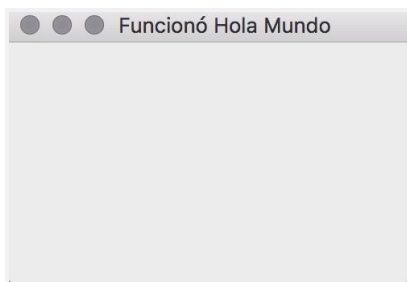
    app = QApplication(sys.argv)

    w = QWidget()
    w.resize(250, 150)
    w.move(300, 300)
    w.setWindowTitle('Funcionó Hola Mundo !!')
    w.show()

    sys.exit(app.exec_())
```

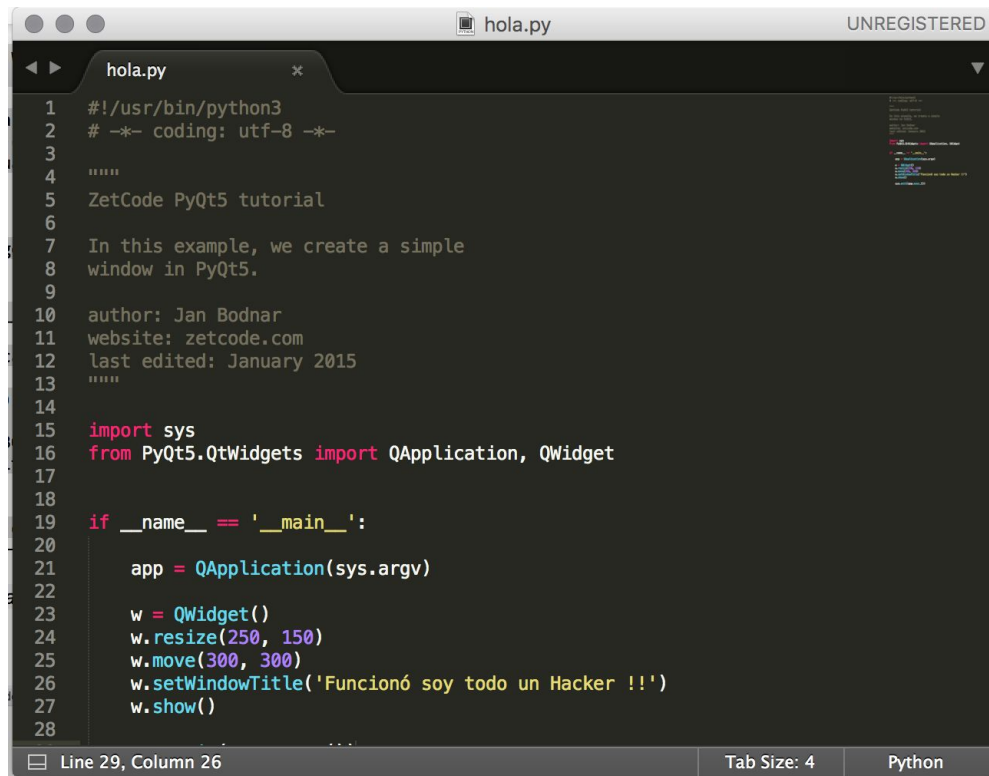
*Ejecutarlo en terminal:*

```
python3 hola.py
```

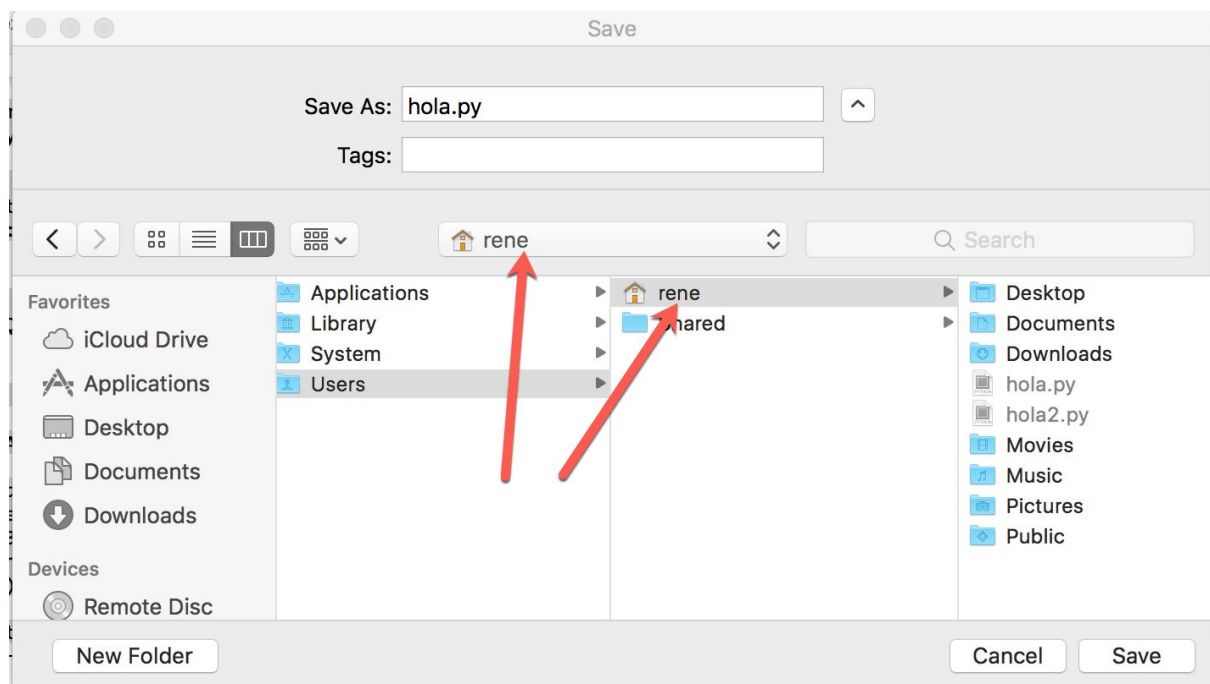




*Sublime sería así:*



```
1  #!/usr/bin/python3
2  # -*- coding: utf-8 -*-
3
4  """
5  ZetCode PyQt5 tutorial
6
7  In this example, we create a simple
8  window in PyQt5.
9
10 author: Jan Bodnar
11 website: zetcode.com
12 last edited: January 2015
13 """
14
15 import sys
16 from PyQt5.QtWidgets import QApplication, QWidget
17
18 if __name__ == '__main__':
19     app = QApplication(sys.argv)
20
21     w = QWidget()
22     w.resize(250, 150)
23     w.move(300, 300)
24     w.setWindowTitle('Funcionó soy todo un Hacker !!!')
25     w.show()
26
27
28
```



*Claro en su directorio de trabajo “rene” en este ejemplo*

*Avanzado, crear su propio APP en MACOS via Python y Py2App*

<https://www.metachris.com/2015/11/create-standalone-mac-os-x-applications-with-python-and-py2app/>

# ***PRIMER PROGRAMA DE PYTHON3+QT5***



***conversionFC.ui***

*Código complementario con esqueleto de python para mandar llamar el formulario en:*

## **PYQT4 CON PYTHON 2.7**

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
# Convierte temperaturas F a C viceversa
# Esqueleto de uso diario
# Por _____
# Solo recordando que este template esta para la versión QT4 y no es funcional QT5
nueva versiones
```

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

```
qtCreatorFile = "conversionFC.ui" # este se cambia por el de Uds.
```

```
Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)
```

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

```
    def __init__(self):
        QtGui.QMainWindow.__init__(self)
        Ui_MainWindow.__init__(self)
        self.setupUi(self)
```

```
    # AQUI VAN LOS BOTONES
```

```
    self.cmdDeCaF.clicked.connect(self.DeCaF)
```

```
    self.cmdDeFaC.clicked.connect(self.DeFaC)
```

```
    self.cmdSalir.clicked.connect(self.Salir)
```

```
# Una cosa son Botones y otra son Funciones, deben ser nombre únicos.
# Cuidar la indentación, es lo parte que exige python.
```

```
# De C a F
```

```
    def DeCaF(self):
```

```
        cel = float(self.txtC.text())
```

```
        fahr = cel * 9 / 5.0 + 32
```

```
        self.txtF.setValue(int(fahr + 0.5))
```

```
# De F a C
```

```
    def DeFaC(self):
```

```
fahr = self.txtF.value()  
cel = ((fahr - 32) * 5) / 9  
self.txtC.setText(str(cel))
```

```
# Cerrar y salir  
def Salir(self):  
    QtGui.qApp.closeAllWindows()
```

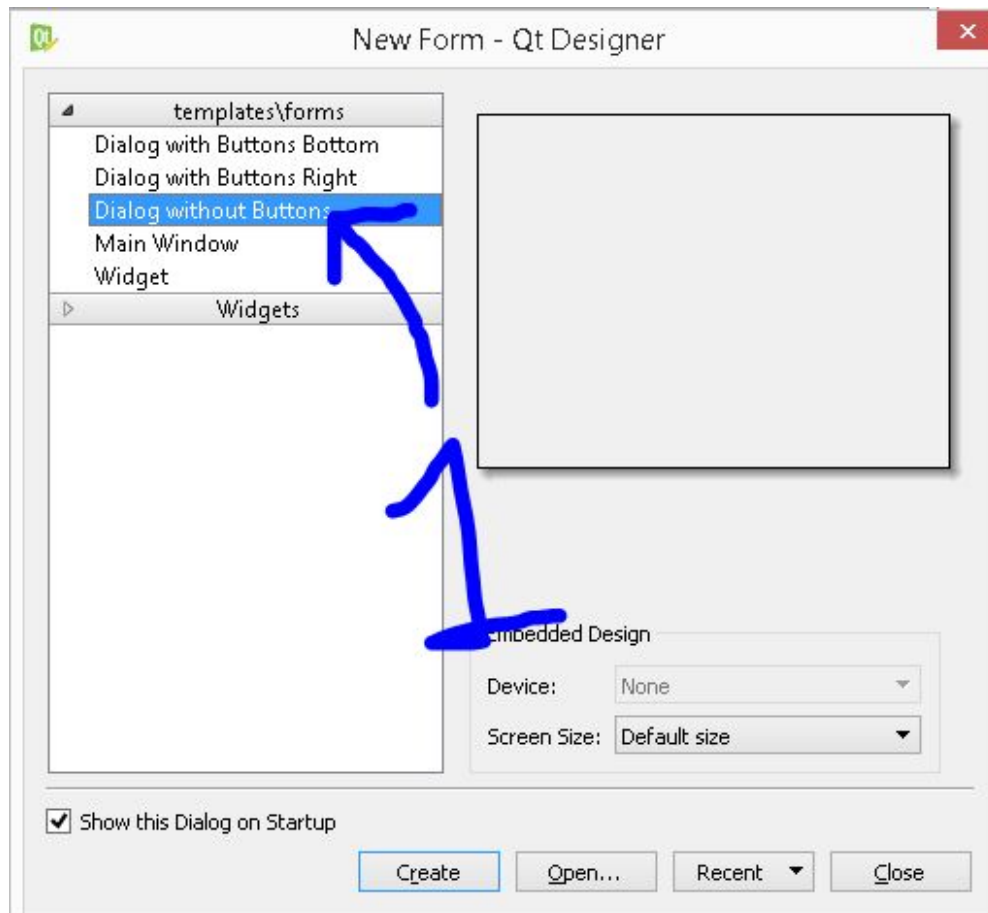
```
if __name__ == "__main__":  
    app = QtGui.QApplication(sys.argv)  
    window = MyApp()  
    window.show()  
    sys.exit(app.exec_())
```

---

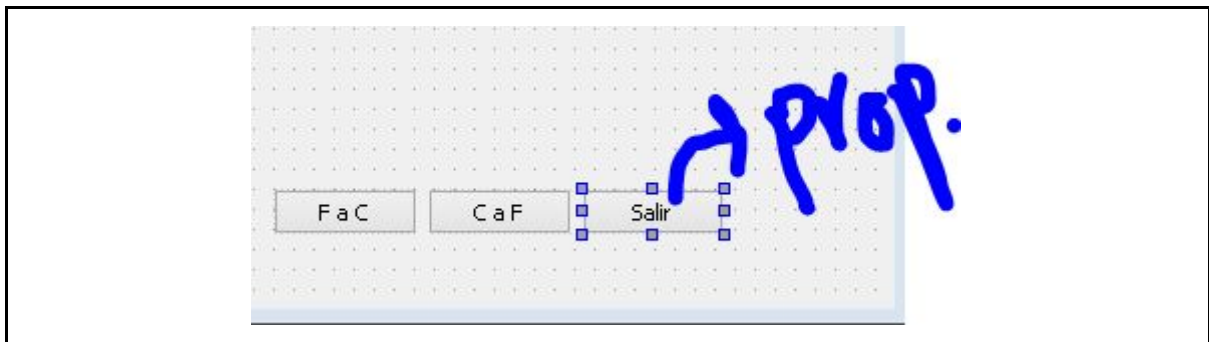
## PYQT5 (PYTHON 3.X)

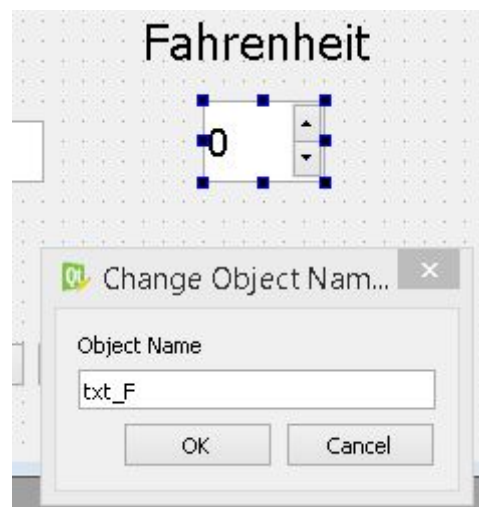
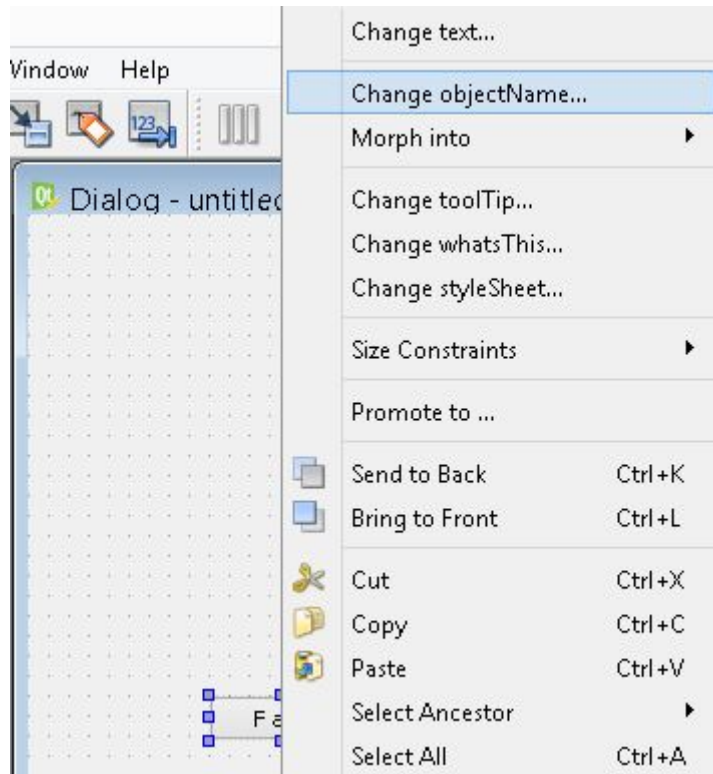
Previa instalación para Windows 7-8-10, iniciamos con QT Designer


### 1- Creación "Dialog without buttons"






### 2.- Arrastrar los botones y cambiar el TEXTO y el nombre del botón:

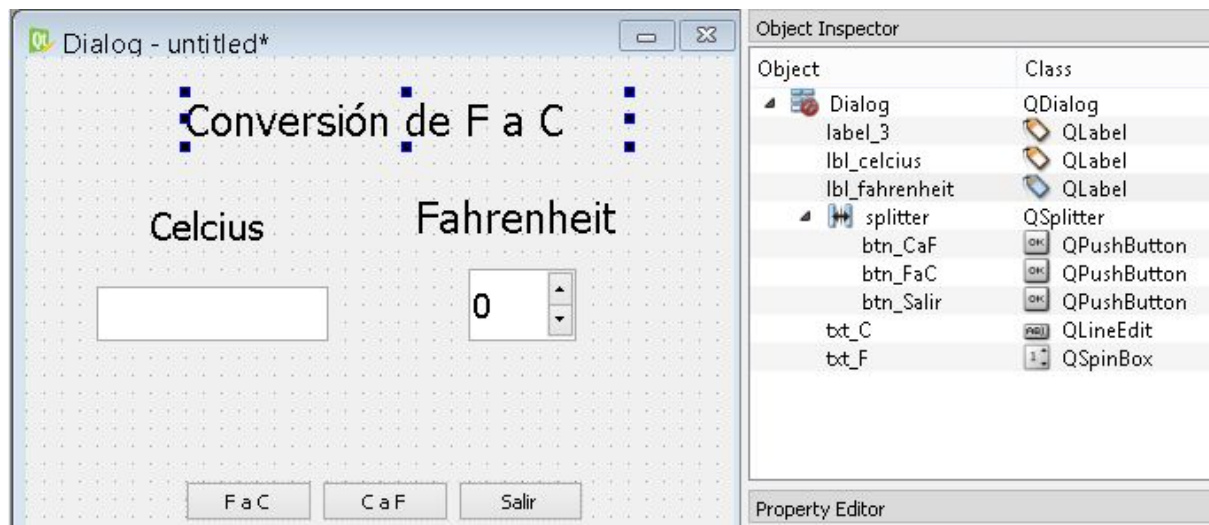




Tipo	Nombre Objeto	Texto	Función a realizar
Botón  Push Button	btn_FaC	F a C	Fahrenheit a Celsius a calcular <div> <div>Fahrenheit to Celsius formula</div> <math display="block">F = C \cdot \frac{9}{5} + 32</math> <div>Celsius to Fahrenheit formula</div> <math display="block">C = (F - 32) \cdot \frac{5}{9}</math> <div>MathATube.com</div> </div>
Botón	btn_CaF	C a F	Celsius a Fahrenheit a calcular

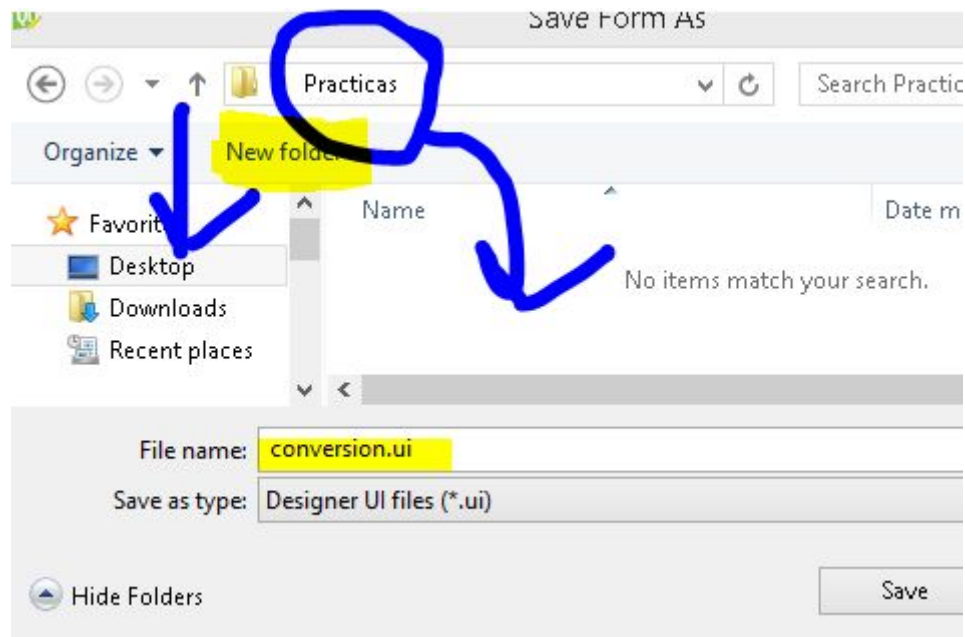
Botón	btn_salir	Salir	Acción de salir de la ventana
Etiqueta (label)  Label	lbl_titulo 1	Conversi on de F a C	Desplegarlo centrado con colores
Etiqueta (label)	lbl_F	Fahrenh eit :	Desplegar Fahrenheit
Etiqueta (label)	lbl_C	Celsius :	Desplegar Celsius
Texto (LineEdit)  Line Edit	txt_C		Aceptar Celsius
Numeros (spinbox)  Spin Box	txt_F		Aceptar Fahrenheit

## RESUMEN DE LOS COMPONENTES EN QT-DESIGNER



## CONVERSIÓN DE DE DE FORMATO \*.UI (XML) A \*.PY (Python3)





- a) En la figura anterior via Explorer de windows se creo la carpeta “practicas” pero tambien hay otras alternativas por comandos
- b) Acceder al directorio de practicas con comando “CMD” del Ejecutar del shell DOS de Windows.
- c) Aplicar nuevo directorio llamado practicas:  
“cd Desktop\practicas”

d) **pyuic5 conversion.ui > conversion.py**

e) Este programa convierte nuestro diseño a Python ejecutable para editarlo a nuestras necesidades, recuerda que cada vez que ejecutes “pyuic5” reemplazará el archivo .PY a la derecha.

```

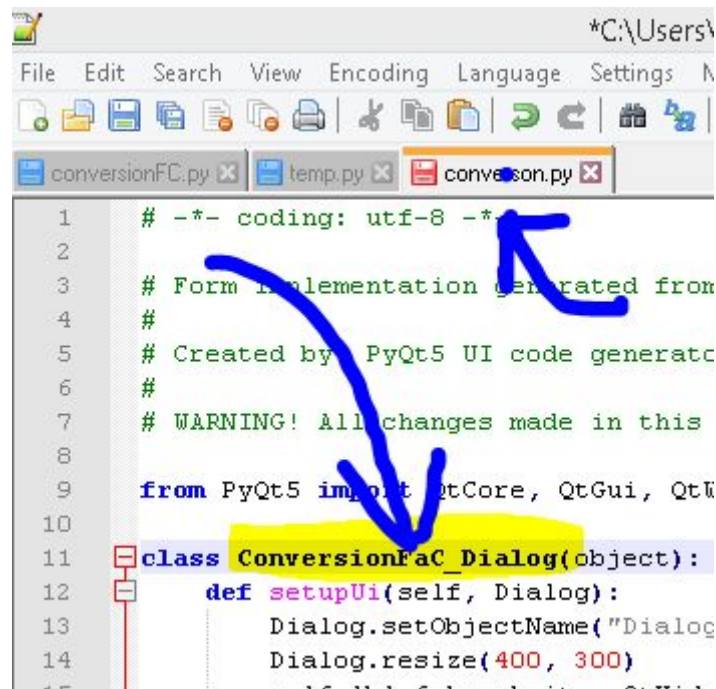
C:\Users\estudiante\Desktop\Practicas>pyuic5 conversion.ui > conversion.py
C:\Users\estudiante\Desktop\Practicas>

```

## CORRECCIONES

Deberemos de hacer una breve corrección para mandar llamar nuestro recién programa convertido.

Abir con Notepadplusplus conversion.py, en el titulo del arreglo del programa renombramos nuestra propiedad “ConversionFaC\_Dialog”



```
1  # -*- coding: utf-8 -*-
2
3  # Form implementation generated from
4  #
5  # Created by PyQt5 UI code generator
6  #
7  # WARNING! All changes made in this
8
9  from PyQt5 import QtCore, QtGui, QtWidgets
10
11 class ConversionFaC_Dialog(object):
12     def setupUi(self, Dialog):
13         Dialog.setObjectName("Dialog")
14         Dialog.resize(400, 300)
```

CORRIDA DEL PROGRAMA Y APLICACIÓN DEL TEMPLATE.

<http://pyqt.sourceforge.net/Docs/PyQt5/designer.html>

App.py

#				
#				
#	IBO 0.			
#				
#				
#	7	8	9	+
#				
#	4	5	6	-
#				
#	1	2	3	x
#				

```

#| | . | 0 | = | | / | |
#| | _ | _ | _ | | _ | |
#| _____ |
#
__author__ = "René Solís"
__copyright__ = "Copyright 2016"
__credits__ = ["PyQT5 Official site"]
__license__ = "GPL"
__version__ = "3.0.0"
__maintainer__ = "René Solís"
__email__ = "rsolis@lazarocardenas.edu.mx"
__status__ = "Production"
__objective__ = "Programa que calcula de F a C"

import sys
from PyQt5.QtWidgets import QApplication, QDialog
from conversion import ConversionFaC_Dialog
# Recuerda "conversion" es el archivo "conversion.py"
# creado con PYUIC5 y lo editamos
# el encabezado ConversionFaC_Dialog
app = QApplication(sys.argv)
window = QDialog()
ui = ConversionFaC_Dialog()
ui.setupUi(window)

window.show()
sys.exit(app.exec_())

```

```
Command Prompt

C:\Users\estudiante\Desktop\Practicas>dir app.py conversion.py
Volume in drive C has no label.
Volume Serial Number is C8AF-8A80

Directory of C:\Users\estudiante\Desktop\Practicas
03/11/2016  01:10 p. m.                913 app.py
Directory of C:\Users\estudiante\Desktop\Practicas
03/11/2016  12:44 p. m.            2,972 conversion.py
                2 File(s)            3,885 bytes
                0 Dir(s)  25,221,001,216 bytes free

C:\Users\estudiante\Desktop\Practicas>
```

Ejecución: `python app.py`



¿Que modificar para que tenga la lógica nuestro programa?  
Solución estará en APP.PY

**¿Y si ocupo corregir el UI pantalla en designer?** Deberá volver a conversion con PYUIC5 y modificar el encabezado de la clase (ConversionFaC\_Dialog) por cada ejecución porque se borra