Instituto Tecnológico de Buenos Aires

Guía de PyQt5

Kammann, Lucas Agustín

Trozzo, Rafael Nicolás

3 de noviembre de 2019

Índice

Comentarios: ¿Qué es esto?

En la guía hay detalles sobre instalación y manejo de comandos, así como descripciones paso a paso y aclaraciones para comprender las cosas un poco más y no cometer errores al momento de usar PyQt5. Si sufre ansiedad de código, probablemente no debería empezar por aquí, sino por los ejemplos.
Se agradece toda colaboración o contribución para con el Repositorio de GitHub, siguiendo el normal flujo de trabajo de Git con Fork y Pull-Request.

Requisitos: ¿Qué necesito para empezar?

- **Python 3:** Hay que instalar Python... fijate acá.
- **PyCharm:** Hay otras opciones como VSCode, Atom, SublimeText, Eclipse, Notepad++, Bloc de notas, del color que quieras... Se recomienda PyCharm, fijate acá.

Nota importante: según el proceso de instalación suele hacerse de forma automática o no, pero puede ser que necesites usar comandos en la consola de Windows ejecutando scripts de Python que son muy usados, como pip principalmente, o el propio intérprete de Python. Si no te funciona, es porque no lo tenes configurado en las variables de entorno del sistema, fijate cómo ¿No te funcionan?.

Figura 1: Error en la ejecución desde consola con PIP

Instalación: ¿Qué tengo que instalar?

- **PyQt5:** Es necesario instalar este paquete. Básicamente es una adaptación del framework QtCreator original empleado en C++ para Python, puede ser que la documentación en este último lenguaje este un poco incompleta. Instalamos PyQt5 ejecutando en consola de comandos de Windows, o en la consola de Linux, el comando:

  pip install PyQt5

- **PyQt5-tools/QtDesigner:** Las herramientas de diseño QtDesginer, entre otras, para usar con este framework hay que instalarlas con otro comando:

  pip install pyqt5-tools

Comandos para consola de windows

¿No te funcionan?

Para que Windows pueda ejecutar un script, bash o algún ejecutable pasando argumentos al programa desde la consola simplementa utilizando una palabra clave, es necesario que esté registrado en las variables de entorno.
Nota importante: si bien abajo explico cómo agregar variables de entorno, es un tema interesante, sirve para otras cosas y tiene vínculo con el mecanismo de Python para importar un paquete o módulo en forma absoluta, podrías mirarme, o a mi, o también a mi.

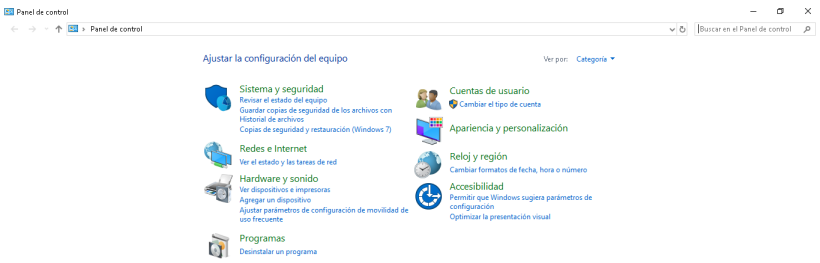

Figura 2: Abrimos Panel de Control de Windows y entramos a Sistema y Seguridad

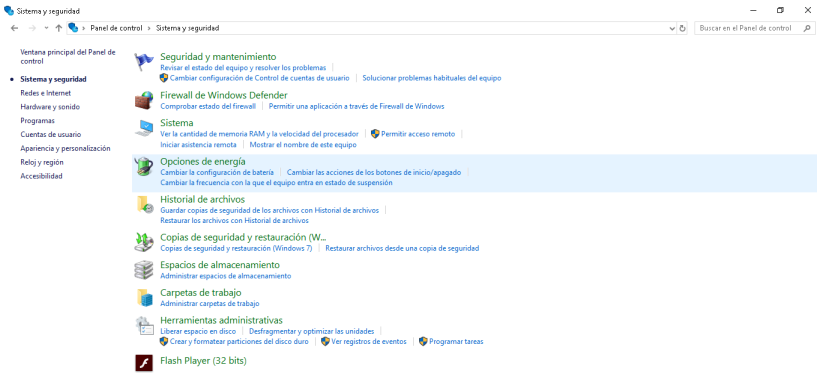

Figura 3: Entramos en Sistema

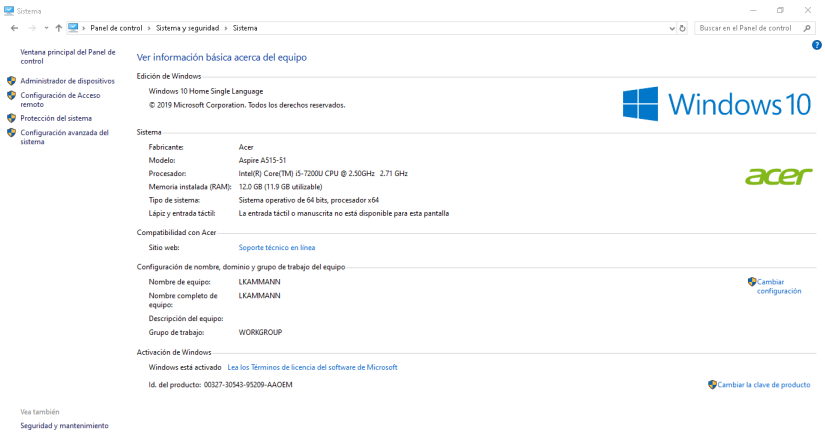

Figura 4: A la izquierda accedemos en Configuración avanzada del sistema

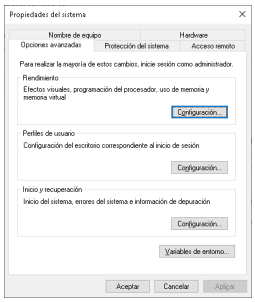

Figura 5: Entramos en el botón que dice Variables de entorno...

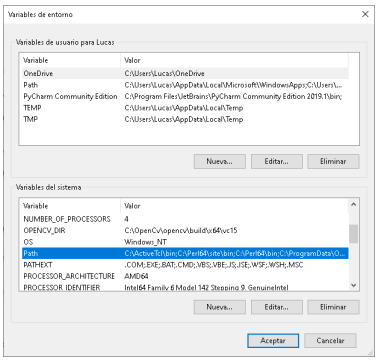

Figura 6: Buscamos en Variables del sistema la que dice Path

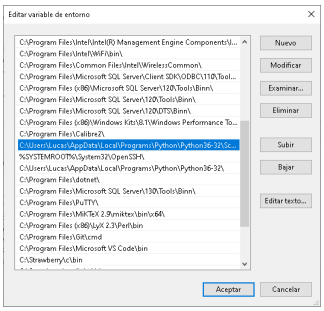

Figura 7: Accediendo a Path aparecen todo su contenido, tenemos que agregar la ubicación de Python y sus scripts

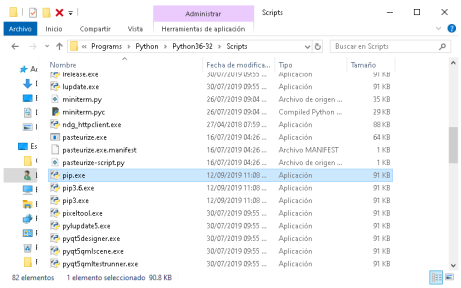

Figura 8: Ejemplo de la ubicación de los scripts de Python

Compilar los diseños de QtDesigner

Con la consola de Windows/Linux abierta, posicionado en la carpeta donde tenemos el archivo "*.ui"que creamos y editamos con el QtDesigner, luego corremos la siguiente línea para compilarlo. Sobre las ventajas o desventajas de este método para utilizar los diseños que realizamos en QtDesigner, es necesario leer otra sección del documento.

pyuic5 -x filename.ui -o output.py

Compilar los recursos de QtDesigner

Con la consola de Windows/Linux abierta, posicionado en la carpeta donde tenemos el archivo "*.qrc"que creamos y editamos con el QtDesigner, luego corremos la siguiente línea para compilarlo.

pyrcc5 filename.qrc -o output.py

QtDesigner: ¿Cómo se usa?

La idea principal de esta sección es orientar en el uso de cada parte de la interfaz de QtDesigner, siguiendo el desarrollo de algunos ejemplos simples que pueden encontrarse también en la carpeta de ejemplos del Repositorio de GitHub.

Creación de un Form

Abrimos QtDesigner, o en New File..,vamos a encontrar esta ventana. Nos deja crear una nueva form de entre esas opciones. Tenemos que tener en mente que **Widget** será cualquier elemento dentro de una interfaz, **Main Window** será una ventana principal que manejará el flujo principal del programa, y **Dialog** es una ventana emergente (¡de hecho esta ventana de QtDesigner es un Dialog!).

- No deberíamos tener más de un Main Window en nuestro proyecto.
- Siempre que creemos entre estas tres opciones, nuestra clase heredará alguna de ellas. QWidget, QMainWindow o QDialog.
- Existen dialogs (QDialog) ya provistos por el framework, fijate después Ventanas de diálogo para reutilizar.

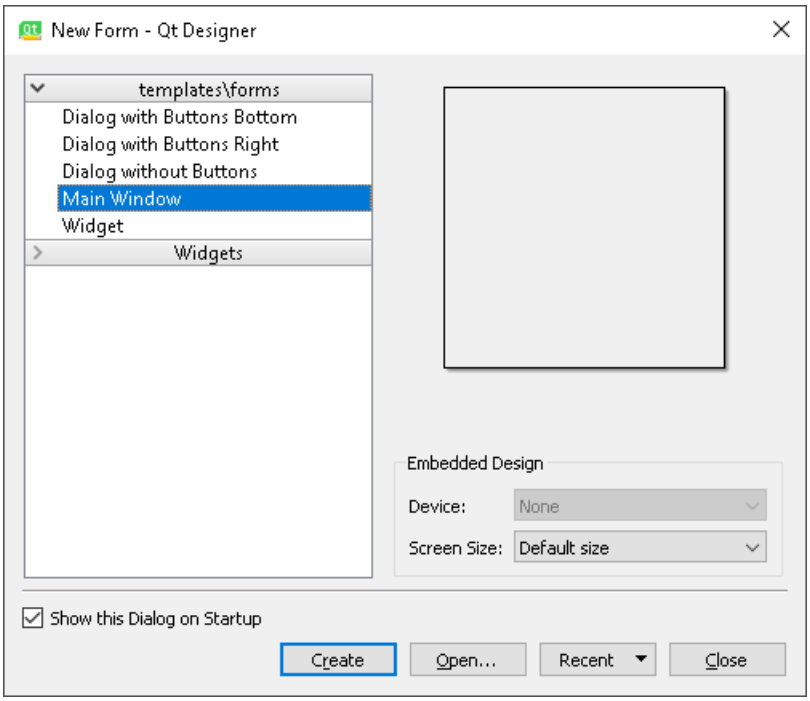

Figura 9: Ventana de creación de formas

BonusTrack: ¿Qué cosas copadas tiene PyQt?

Ventanas de diálogo para reutilizar

Existen algunos diálogos ya creados por el framework de Qt para que reutilicemos, como el **QFileDialog**, **QMessageBox**, **QColorDialog**, **QInputDialog**, **QProgressDialog**, entre otros... y su utilización es muy sencilla. Observar en la Fig. 10.
Se los puede utilizar configurandolos con el constructor, instanciándolos y luego llamando a alguno de sus métodos para hacerlos visibles y luego recuperar los datos que obtuvieron. Por otro lado, se pueden ejecutar desde alguno de sus métodos estáticos.

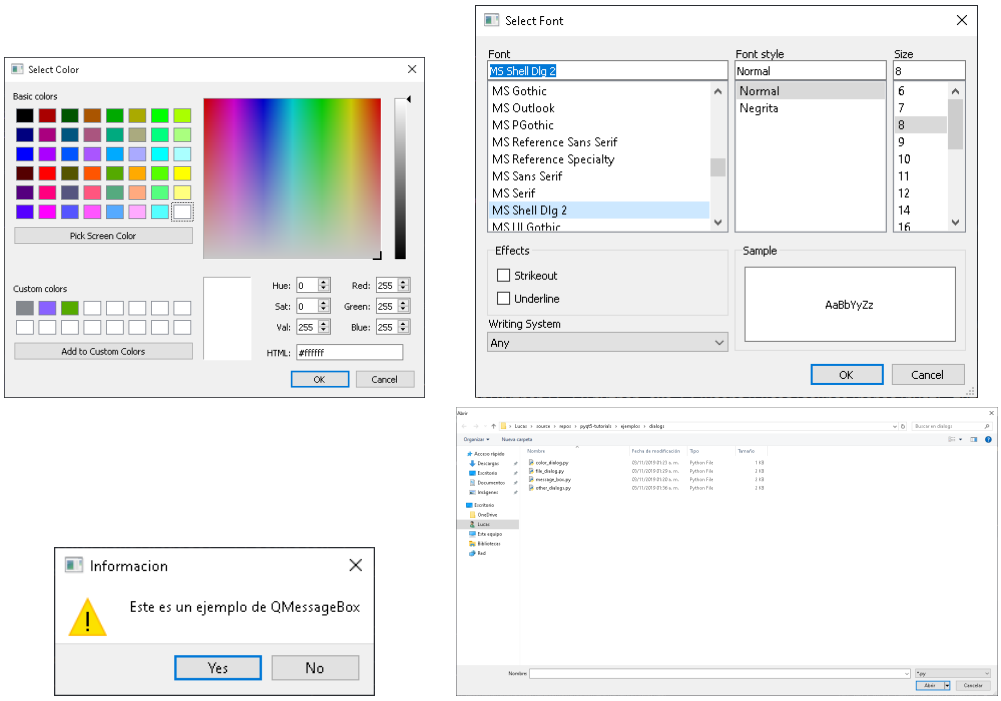

Figura 10: Ejemplos de QColorDialog, QFileDialog, QFontDialog y QMessageBox

PyQt

¿Cómo estructurar los proyectos?

Los proyectos que involucran el diseño de una GUI utilizando PyQt5 estarán compuestos por el Backend y el Frontend, conceptos que vienen del desarrollo web, y podes ver acá. Entonces, tu proyecto va a tener código en Python de la GUI, de la lógica de lo que hagas, y otros archivos como recursos, stylesheets, imagenes, etc... se propone la siguiente estructura de archivos muy utilizada en PyQt5, y Python en algunos casos: