**INTEGRACIÓN PyQtGRAPH CON QtDESIGNER**

Integrar elementos de visualización e interfaces gráficas se suele realizar por medio de la integración de los paquetes de *PyQT* y *Matpotlib.* Sin embargo, existe un paquete que tiene ciertas funcionalidades de base que hace relevante el conocerlo. Este paquete se llama *PyQtGraph*, y durante el desarrollo de este ejercicio se observará cómo integrar diferentes elementos que este ofrece, a QtDesigner, observando también cómo acceder a la funcionalidad de estos elementos.

Para verificar que los conceptos mencionados quedaron claros se pide implementar una interfaz gráfica que permita al usuario ingresar un número y al oprimir un botón este se grafique, a medida que se vayan agregando más números se debe ir generando una curva en el objeto donde se están graficando los números. Se debe verificar que el usuario sí ingrese un valor numérico, en caso de no hacerlo debe aparecer una ventana emergente que alerte al usuario sobre la equivocación.

La interfaz gráfica puede tener la siguiente forma:

Graphical user interface

Description automatically generated with low confidence

**Solución**

Para integrar un objeto perteneciente a PyQTGraph a una interfaz gráfica por medio de QtDesigner se deben seguir los siguientes pasos:

1. Implementar un Graphics View disponible en la sección Display Widgets:

Graphical user interface, application, Word

Description automatically generated

1. Dar click derecho y seleccionar la opción “Promote to …”

Graphical user interface, application

Description automatically generated

1. En este paso se decide qué clase será la que se utilizará en la interfaz gráfica. Dentro de las opciones de clases que hay disponibles en pyqtgraph están: “PlotWidget”, “GraphicsLayoutWidget”, etc. El nombre de la clase decidida se ingresa en Promoted class name y en Header file se ingresa “pyqtgraph”. Luego se da click en “Add” y luego en “Promote”.

Graphical user interface, application

Description automatically generated

Luego de hacer esto, según el nombre que se le haya dado al atributo, este podrá ser empleado en la lógica de la interfaz gráfica llamando los métodos de la clase.

Para la resolución del ejercicio planteado se implementó la siguiente interfaz gráfica en Qt Designer:

Graphical user interface, application

Description automatically generated

En este caso, como se puede ver en el campo de los objetos, la clase promovida fue el PlotWidget. O sea que este fue el nombre que se ingreso en el campo Promoted class name. La interfaz gráfica ejecutada se observa de la siguiente forma:

Graphical user interface, Excel

Description automatically generated

Antes de comenzar con la parte funcional de la interfaz gráfica se mencionará cómo se puede garantizar que cuando se oprima el botón de maximizar la interfaz se maximise automáticamente. Cuando se oprime el botón maximizar en la interfaz gráfica, por defecto los elementos tienden a pertenecer donde se encuentran:

A picture containing shape

Description automatically generated

Para que los objetos respondan al botón de maximizar en Qt Designer se debe dar click derecho sobre el objeto padre de toda la ventana, en este caso QWidget, en el caso de una ventana principal sería QMainWindow. Y luego dar click en Lay out y luego dar click sobre Lay Out Vertically.

Graphical user interface, application, Word

Description automatically generated

Luego de realizar esta configuración:

A picture containing shape

Description automatically generated

**Solución ejercicio:**

Para la solución del ejercicio, luego de configurar la interfaz en Qt Designer, se utilizó la siguiente lógica:

|  |  |
| --- | --- |
|  | En este caso hay ciertas líneas de código que siempre suelen emplearse cuando se crean interfaces gráficas con PyQt. Estas líneas están señaladas en verde, donde las primeras solo son paquetes que se importan (Varían dependiendo de la aplicación) y las líneas finales, que siempre se utilizan ya que son las que ejecutan la ventana. En el caso de las líneas finales solo cambia cuando se instancia la ventana, cuyo objeto se almacena en la variable main.  Con respecto a las líneas señaladas de azul verdoso, estas son usadas para configurar los colores de fondo y de dibujado de cualquier objeto disponible en Pyqtgraph. En este caso se elige que el fondo sea de color blanco y el color de dibujado de negro. Por defecto los colores están trocados.  La lógica en sí de la interfaz gráfica se encuentra señalada en amarillo. En el método constructor se carga la interfaz diseñada en Qt Designer, se genera un atributo llamado contenido, el cual se encargará de almacenar los valores ingresados por el usuario y por último se asocia el método graficar al evento de dar click sobre el atributo botón.  En el método graficar se emplea una estructura try para manejar las posibles excepciones que pueden generarse debido a que el usuario no ingrese un valor numérico. Dentro del try se agrega al atributo contenido el entero que haya en el campo, este valor se captura por medio del método text(). Luego, se grafica en el PlotWidget por medio del método plot(), el cual recibe también arrays de Numpy como argumentos. Para más información de estos métodos se recomienda ir a la documentación del paquete de Pyqtgraph.  Si se genera alguna clase de excepción se mostrará una ventana de alerta con el mensaje: Ingrese valores numéricos. La generación de esta ventana se realiza por medio de la clase QMessageBox, la cual está disponible en QtWidgets. Para configurar el mensaje que se mostrará se usa el método setText() y para ejecutarla se emplea el método exec\_().  A continuación, se muestra una serie de imágenes que muestra la funcionalidad de la interfaz: |

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

En la última imagen se muestra una de las funcionalidades que traen por defecto los elementos de Pyqtgraph, y es que se puede interactuar con los gráficos por medio del mouse, haciendo zoom si es necesario, o moviéndose en algo uno de los ejes.