INSTRUMENTACION VIRTUAL

RADIO BUTTON

El Radio Button es un widget muy usado, este widget nos sirve para agregar una lista de opciones y seleccionar únicamente una. Cuando una opción es seleccionada la anterior es deseleccionada, cada Radio Button tiene la opción de personalizar su campo *text*. Si se desea tener varios grupos independientes de Radio Buttons lo que se tiene que hacer es poner cada grupo de Radio Buttons dentro de un contenedor como son los Layouts (vertical, horizontal, grid o form). El Radio Button es instancia de la clase QRadioButton y los métodos que soporta son los siguientes:

METODOS.

- **isChecked()**. Este método retorna un estado True si el radio button es seleccionado.
- **setIcon()**. Este método despliega un icono con el radio button.
- **setText()**. Este método permite cambiar el texto al radio button. Si se desea agregar un atajo por teclado se tiene que agregar el carácter (&) detrás de la letra que usaremos como atajo.
- **setChecked()**. Este método nos sirve para establecer un radio button como seleccionado por default, esto se hace pasando el valor True a este método.

SEÑALES.

Las señales emitidas por QRadioButton son las siguientes:

- **toggled()**. Esta señal es emitida cuando el radio button cambia de seleccionado a deseleccionado o viceversa.
- **clicked()**. Esta señal es emitida cuando el radio button es activado (haciendo click sobre el radio button) o cuando su atajo por teclado es activado.

Para que esto quede más claro vamos a hacer un ejemplo.

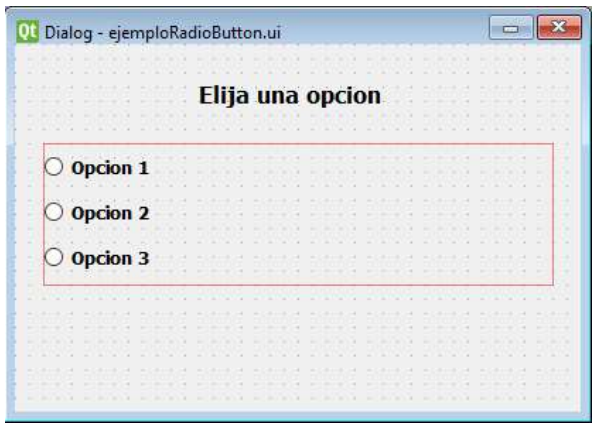
Vamos a abrir el QtDesigner y creamos un Dialogo sin botones y realizamos los siguientes pasos:

1. Agregamos dos Label y tres Radio Button dentro de la ventana.
2. Cambiemos la propiedad *text* del primer label por el siguiente texto “Elija una opción” y borramos el texto del segundo label en su propiedad *text*.
3. Cambiamos en la propiedad *text* de cada Radio Button por los siguientes textos, “Opcion 1”, “Opcion 2” y “Opcion 3”.
4. En la propiedad *objectName* de cada Radio Button borramos y ponemos lo siguiente. rbOpcio1, rbOpcion2, rbOpcion3 y también en el label2 cambiamos su nombre por lblResultado.
5. Guardamos la aplicación con el nombre radioButton.ui.

6. Convertimos el archivo ui en código Python con el siguiente comando.

Pyuic5 radioButton.ui -o radioButton.py

La interfaz tendría la siguiente forma en vista previa en QtDesigner.



El último paso es introducir el siguiente código en el PyCharm para esto creamos un proyecto nuevo vacío y cuando nos pida la ruta donde deseamos guardar el proyecto tiene que ser en la misma carpeta donde tenemos el archivo que convertimos a código python. El código es el siguiente.

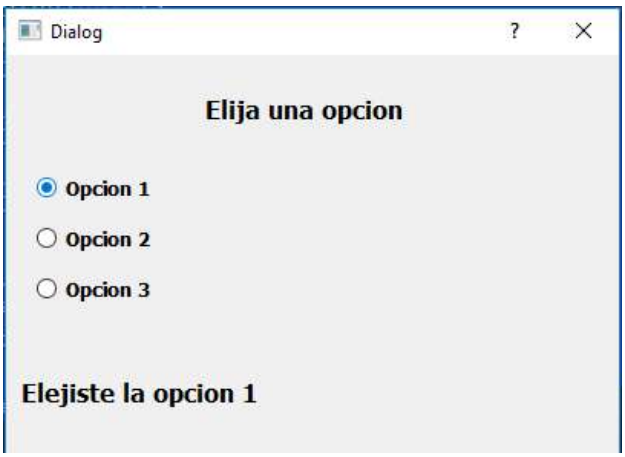
```
import sys
from PySide6.QtWidgets import QApplication, QDialog
from radioButton import Ui_Dialog


class MiFormulario(QDialog, Ui_Dialog):  1 usage
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.rbOpcion1.toggled.connect(self.mensajeOpcion)
        self.rbOpcion2.toggled.connect(self.mensajeOpcion)
        self.rbOpcion3.toggled.connect(self.mensajeOpcion)

    def mensajeOpcion(self):  3 usages
        if self.rbOpcion1.isChecked():
            self.lblResultado.setText("Elejiste la opcion 1")
        elif self.rbOpcion2.isChecked():
            self.lblResultado.setText("Elejiste la opcion 2")
        elif self.rbOpcion3.isChecked():
            self.lblResultado.setText("Elejiste la opcion 3")
```

```
if __name__ == "__main__":
    app = QApplication(sys.argv)
    w = MiFormulario()
    w.show()
    sys.exit(app.exec())
```

Como resultado tenemos:



Vemos que si vamos seleccionando cada Radio Button nos muestra la opción que hemos elegido.