1. Wprowadzenie

Pliki .ui są często tworzone za pomocą narzędzi do projektowania interfejsów graficznych, takich jak **Qt Designer**. Zawierają one informacje o układzie elementów na formularzu. Pliki .py to pliki z kodem Python, które zawierają logikę aplikacji. Aby połączyć te dwa rodzaje plików i stworzyć interaktywne aplikacje, należy przekonwertować plik .ui na kod Pythona i następnie dodać do niego obsługę zdarzeń.

2. Konwersja pliku .ui na kod Pythona

Istnieje kilka sposobów na przekonwertowanie pliku .ui na kod Pythona. Najpopularniejszym jest użycie narzędzia **pyuic5**.

Aby zainstalować narzędzie **pyuic5** musimy w wierszu poleceń wpisać:

pip install PyQt5

```
C:\Users\Mario>pip install PyQt5
Requirement already satisfied: PyQt5 in c:\python312\lib\site-packages (5.15.11)
Requirement already satisfied: PyQt5-sip<13,>=12.15 in c:\python312\lib\site-packages (from PyQt5) (12.15.0)
Requirement already satisfied: PyQt5-Qt5<5.16.0,>=5.15.2 in c:\python312\lib\site-packages (from PyQt5) (5.15.2)

[notice] A new release of pip is available: 24.2 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Aby przekonwertować plik .ui na kod Pythona piszemy w wierszu poleceń oraz katalogu w którym znajduje się nasz plik polecenie:

pyuic5 -x nazwa_pliku.ui -o nazwa_pliku_ui.py

Poniżej zrzut ekranu z utworzenia konwersji:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\Mario\Desktop\2. Scenariusze lekcji\4.PROGRAMO
y\egzamin>pyuic5 -x okno.ui -o okno.py

C:\Users\Mario\Desktop\2. Scenariusze lekcji\4.PROGRAMO
y\egzamin>
```

3. Dodawanie obsługi zdarzeń przycisków

Po przekonwertowaniu pliku .ui, otrzymasz plik Pythona, który zawiera definicję wszystkich elementów interfejsu.
Poniżej widok pliku .py:

okno.py

```python
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'okno.ui'
#
# Created by: PyQt5 UI code generator 5.15.11
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):  1 usage
    def setupUi(self, MainWindow):  1 usage
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(557, 279)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.gridLayout = QtWidgets.QGridLayout(self.centralwidget)
        self.gridLayout.setObjectName("gridLayout")
        self.layout_main = QtWidgets.QVBoxLayout()
        self.layout_main.setContentsMargins(10, 10, 10, 10)
        self.layout_main.setObjectName("layout_main")
        self.layout_main_columns = QtWidgets.QHBoxLayout()
```

Aby dodać obsługę zdarzeń przycisków, należy utworzyć klasę głównego okna, która dziedziczy po klasie z wygenerowanego pliku i dodajemy własną logikę.

Tworzymy w naszym projekcie nowy plik pythona o nazwie np. **main.py**, importujemy moduły, ładujemy interfejs i tworzymy klasę głównego okna aplikacji.

Poniżej widok pliku main.py :

okno.py | main.py | okno.ui

```
1   import sys
2   from PyQt5 import QtCore, QtGui, QtWidgets
3   from okno import Ui_MainWindow
4   class MainWindow(QtWidgets.QMainWindow, Ui_MainWindow):
5       def __init__(self):
6           super(MainWindow, self).__init__()
7           self.setupUi(self)
8           # Tutaj dodajemy obsługę zdarzeń
9
10          # Tutaj dodajemy metody do łączenia zdarzeń
11
12  if __name__ == "__main__":
13      app = QtWidgets.QApplication(sys.argv)
14      window = MainWindow()
15      window.show()
16      sys.exit(app.exec_())
```

4. Uruchomienie okna

Po wpisaniu kodu można uruchomić interfejs utworzony w Qt Designer.

Wyjaśnienie kodu:

- Linie 1-2: Importujemy niezbędne moduły.
- Linia 3: Importujemy interfejs z wygenerowanego pliku Pythona.
- Linia 4: Tworzymy klasę MainWindow, która dziedziczy po QtWidgets.QMainWindow i Ui_MainWindow.
- Linia 7: Wywołujemy metodę setupUi z klasy bazowej, która inicjalizuje interfejs.
- Linia 12: Instrukcja „if __ name __ == '__ main __'" w Pythonie sprawdza, czy bieżący skrypt jest uruchamiany bezpośrednio jako program główny, czy jest importowany jako moduł do innego programu.
- Linia 13-16: Instrukcje uruchamiające aplikację

Ważne:

Podczas połączenia zdarzeń upewnij się, że nazwa widżetu w kodzie Pythona jest taka sama jak w pliku .ui.