***Comandos Básicos***

Criamos a aplicação da Interface com:  
  
QApplication([])

Criamos uma Janela com QWidget e adicionamos um Layout para essa janela em específico, como: QVBoxLayout para uma Caixa VERTICAL

Podemos criar um Rótulo ou então uma caixa de texto com QLabel e um botão com QPushButton

Ao criar as janelas e seus elementos separados, juntamos tudo com:  
  
Variavel que foi definido o layout, nesse exemplo: layout.

Layout.addWidget(Nome do elemento a ser adicionado)

Configurar o layout na janela:

Janela.setLayout(layout)

Exibimos a janela: janela.show()

Executamentos a aplicação: app.exec()

***Eventos e Interatividade (Conectando Botões)***

Eventos e Interatividade (Conectando Botões)

Para ativar eventos precisamos criar uma função referente ao evento que será ativado.

Exemplo de clicar em um botão e alterar um texto

Definimos uma função (cliquei\_no\_botao())

E colocamos o seguinte comando:

Def cliquei\_no\_botao():

Label.setText(“Você clicou no botão!”)

App = QApplication([]) # Definimos a aplicação

Janela = QWidget() # Criamos a janela

Janela.setWindowTitle(“Interatividade”) # Definimos o nome da janela

Layout = QVBoxLayout() # Definimos o tipo do layout

Label = QLabel(“Clique no botão”) # Esse será o texto a ser alterado na função pois usamos o NomeVariavel.setText(“...”) apenas alterando o texto mostrado

Botão = QPushButton(“Clique Aqui”)

Botão.clicked.connect(cliquei\_no\_botao) # Conecta a função de trocar o texto com o evento de clicar no botão

Layout.addWidget(label)

Layout.addWidget(botao)

Janela.setLayout(layout)

Janela.show()

App.exec()

***Criando Janelas Personalizadas***

Criamos uma classe que Herda QWidget:

Class MinhaJanela(QWidget)

Def\_\_init\_\_(self) # definimos uma função \_\_init\_\_

Super().\_\_init\_\_()

Segue normalmente, o único porém é: apenas colocamos o self.nome da variável

Também podemos colocar, dentro da classe, funções que dizem respeito aquela janela, como:

Def mudar\_texto(self):

Self.label.setText(“Texto alterado!”)

***Criando Caixas de Texto (QLineEdit)***

Aqui podemos utilizar a mesma estrutura anterior de classe.

Com o PySide6 podemos adicionar caixas de Inputs para o usuário com o comando:

**QLineEdit()**

Para pegar esse valor que foi digitado fazemos da seguinte forma:

Variável = self.variavel\_caixa\_texto.text()

Podemos utilizar isso para pegar o valor escrito no input oferecido ao usuário

***Criando Tabelas (QTableWidget)***

Para criar tabelas utilizamos o comando:  
QTableWidget(Número de linhas, Número de Colunas)

Adicionar o nome das colunas (cabeçalhos):

Self.variavel\_da\_tabela.setHorizontalHeaderLabels([“...”,”...”]) # Se coloca o numero de valores de acordo com o numero de colunas definidas anteriormente, sempre em formato de lista “[]”

Para definir valores utilizamos o seguinte comando formatado como:

Self.variavel\_da\_tabela.setItem(LinhaX, ColunaY, QTableWidgetItem(“Valor a definir”))

***Criando Janelas com Múltiplas Telas***

Para alternar entre Janelas, primeiro precisamos criar 2 janelas distintas, fazemos isso criando duas classes distintas com um novo argumento além do self, Stack.

Exemplo: class Tela1(QWidget):

Def \_\_init\_\_(self, stack):

E:

class Tela2(QWidget):

Def \_\_init\_\_(self, stack):

Em ambos precisamos colocar uma linha para identificar cada uma e como vão se “conversar”.

Utilizamos a linha de evento:

Self.botao.clicked.connect(lambda: stack.setCurrentIndex(1)) para tela1

E

Self.botao.clicked.connect(lambda: stack.setCurrentIndex(0)) para tela2

Ou seja, quando ativar o botão, desencadeara o evento para mudar de tela.

No final, precisamos adicionar algumas coisas:

App = QApplication([])

Stack = QStackWidget() # Permiti alterar entre as Telas

Stack.addWidget(Tela1(Stack))

Stack.addWidget(Tela2(Stack))

Stack.show()

App.exec()