Les interfaces graphiques

Les interfaces graphiques (ou interfaces homme machine) sont appelées GUI (Graphical User Interface).

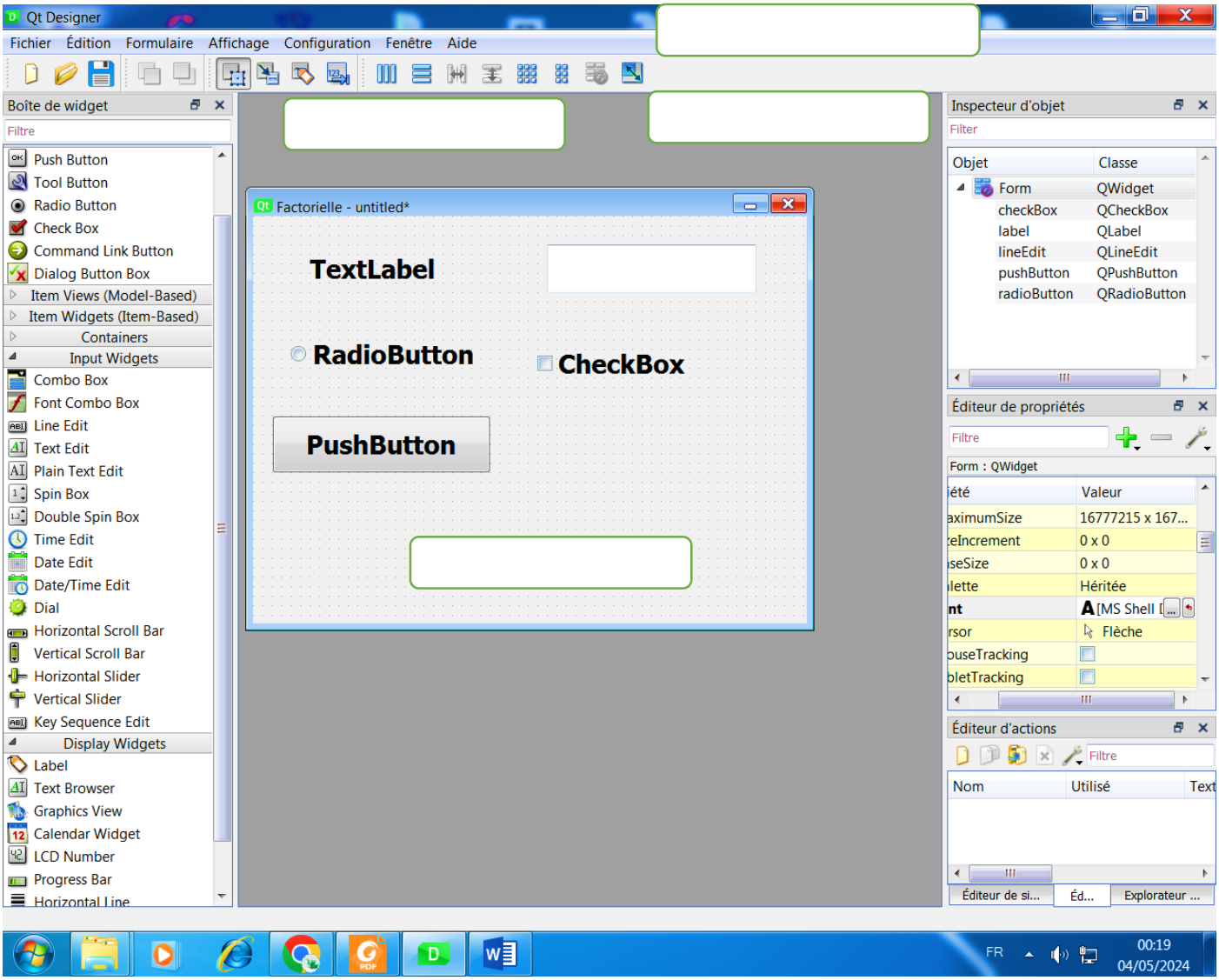
Elles permettent à l'utilisateur d'interagir avec un programme informatique, grâce aux différents objets graphiques (zone de texte, case à cocher, bouton radio, bouton poussoir, menu, …).

Ces objets graphiques sont généralement actionnés avec un dispositif de pointage, le plus souvent la souris.

Dans le présent Cours on utilisera :

- Qt Designer pour créer des interfaces Graphiques et
- Python pour les programmer.

Espace de travail : QtDesigner



Activité :

En utilisant l'éditeur de Python disponible sur l'ordinateur, ouvrir le fichier« ***Factorielleinterface.py*** » existant dans le dossier « ACTIVITE » situé sur le lecteur **D:** l'exécuter puis dégager son rôle.

Le code python du programme « ***Factorielleinterface.py*** » est présenté dans le tableau ci-dessous.

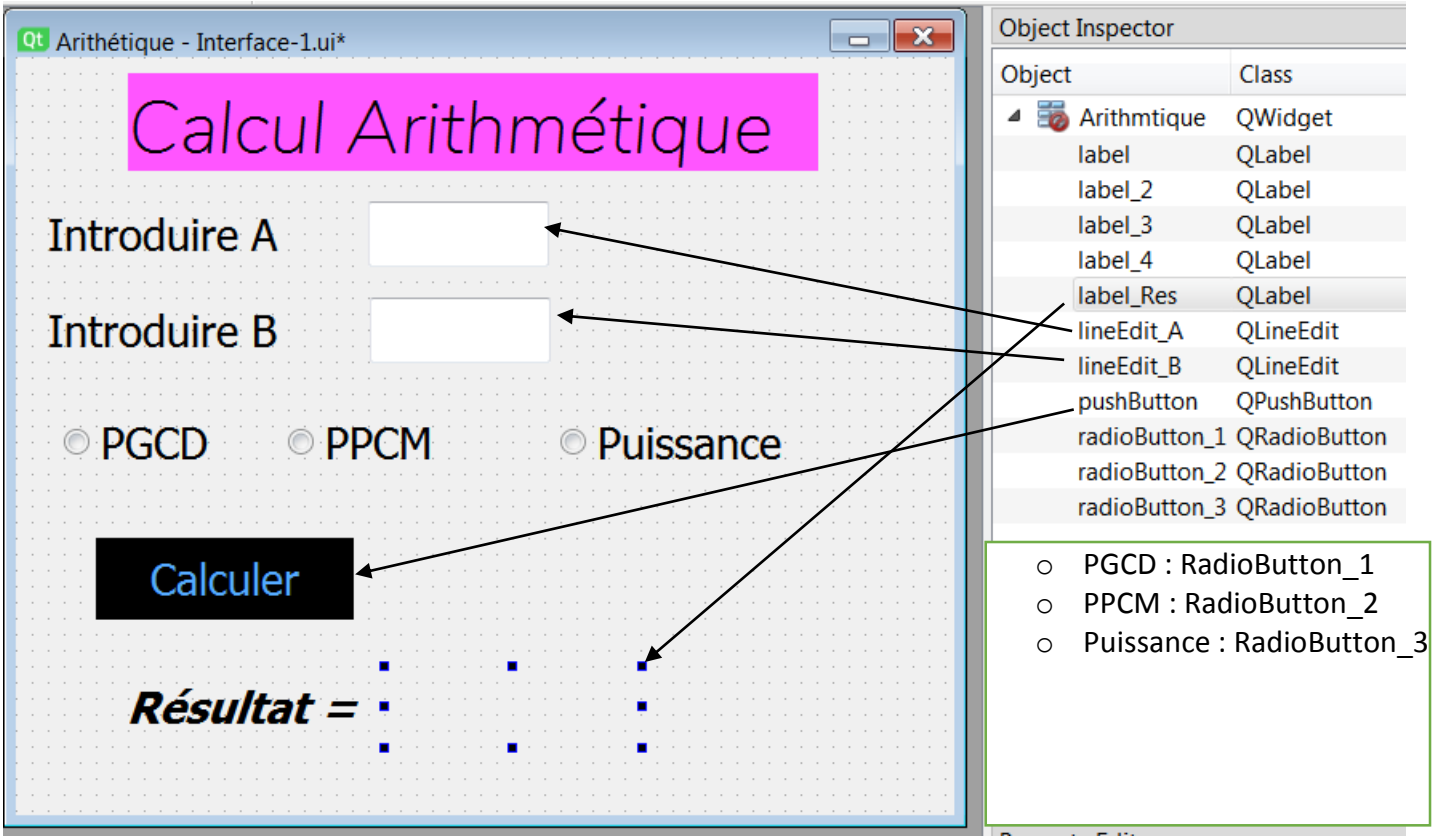
Remplir la colonne rôle par les rôles de la liste suivante :

/ Événement sur le bouton / Lecture d'un fichier externe avec l'extension « ui » / Bibliothèques /
/ Création de l'application graphique / Fonction écrite par l'utilisateur /
/ Affichage de l'interface graphique /

Bloc de code	Rôle
from PyQt5.uic **import** loadUi **from** PyQt5.QtWidgets **import** QApplication	
def *factorielle* ( ) : N = **int** (fenetre.Nb.**text** ( )) f = 1 **for** i **in range** (2 , N+1) : f = f * i fenetre.Res.**setText** ("La factorielle = " **+ str** (f))	
app = **QApplication** ([ ])	
fenetre = **loadUi** ("interfacefact.ui")	
fenetre . **show** ( )	
fenetre.btcalcul.**clicked.connect** (*factorielle*)	
app.exec_ ( )	

Application 1 : Lancer QtDesigner

1- créer l'interface suivante :



2- Renommer les Widgets comme indiqué dans la figure
3- Enregistrer le fichier sous le nom Interface-1.ui dans D:\APPLICATION_1
4- Les widgets utilisés :
- Label : étiquette
- LineEdit : rdt une zone de saisie
- RadioButton : Bouton Radio
- PushButton : bouton sensible au clic

5- Nous voulons programmer cette interface de la façon suivante :
   a. L'utilisateur saisi deux valeurs dans les deux zone de saisie
   b. Ensuite il coche un des 3 boutons radios puis
   c. Lorsqu'il clique sur le bouton Calculer le calcul se fait et le programme affiche le résultat dans la zone label_Res

6- Lancer Thonny et créer un nouveau fichier
7- Cliquer sur le bouton « **Ajouter code PyQt5** » qui permet d'insérer un script Python permettant d'appeler le fichier (**D:APPLICATION_1)« *Interface-1.ui* »**, l'enregistrer sous le nom « ***arithmétique.py*** »
8- Dans ce programme on a besoin de 3 fonctions: calcul PGCD, calcul PPCM et calcul puissance
9- Ouvrir le fichier source.txt puis copier ces trois fonctions dans votre code
10- Compléter votre programme puis l'exécuter

```
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
def pgcd(a,b):
    while (a!=b):
        if a>b:
            a=a-b
        else:
            b=b-a
    return a
def ppcm(a,b):
    return a*b/pgcd(a,b)
def puissance(a,b):
    p=1
    for i in range(b):
        p=p*a
    return p
def pushButton_click():
    a=int(fenetre.lineEdit_A.text())
    b=int(fenetre.lineEdit_B.text())
    if fenetre.radioButton_1.isChecked() :
        r=pgcd(a,b)
    elif fenetre.radioButton_2.isChecked() :
        r=ppcm(a,b)
    else :
        r=puissance(a,b)
    fenetre.label_Res.setText(str(r))
app = QApplication([])
fenetre = loadUi ("D:/APPLICATION_QT/APPLICATION_1/Interface-1.ui")
fenetre.show()
fenetre.pushButton.clicked.connect ( pushButton_click )
app.exec_()
```

Zone de texte : « QLAbel » :

Objet (Widget)	Méthode	Rôle
Label fenetre.nom_label.	text ( )	Permet de récupérer le texte existant dans un objet «label»
	setText (val)	Permet d'initialiser le texte de l'objet «label» à une valeur
	clear ()	Permet d'effacer le contenu de l'objet «label»

Zone de saisie : « QLineEdit »

Objet (Widget)	Méthode	Rôle
Line Edit fenetre.nom_line edit.	text ( )	Permet de récupérer le texte existant dans un objet « Line Edit »
	setText (val)	Permet d'initialiser le texte de l'objet«Line Edit» à une valeur
	setFocus ()	Mettre le curseur dans la zone de saisie
	clear ()	Permet d'effacer le contenu de l'objet «Line Edit»

Bouton poussoir : « QPushbutton » :

Objet (Widget)	Evénement	Rôle
Push Button fenetre.nom_Bouton.	clicked ( )	Déclencher quand le bouton est cliqué
	pressed ( )	Déclencher quand le bouton est pressé
	released ( )	Déclencher quand le bouton est relâché
	toggled ( )	Déclencher quand l'attribut «checkable = Oui»

Exemple : **Associer un signal (événement) à un slot (Module)**

fenetre.nom_Bouton.clicked.connect (Nom_Module)

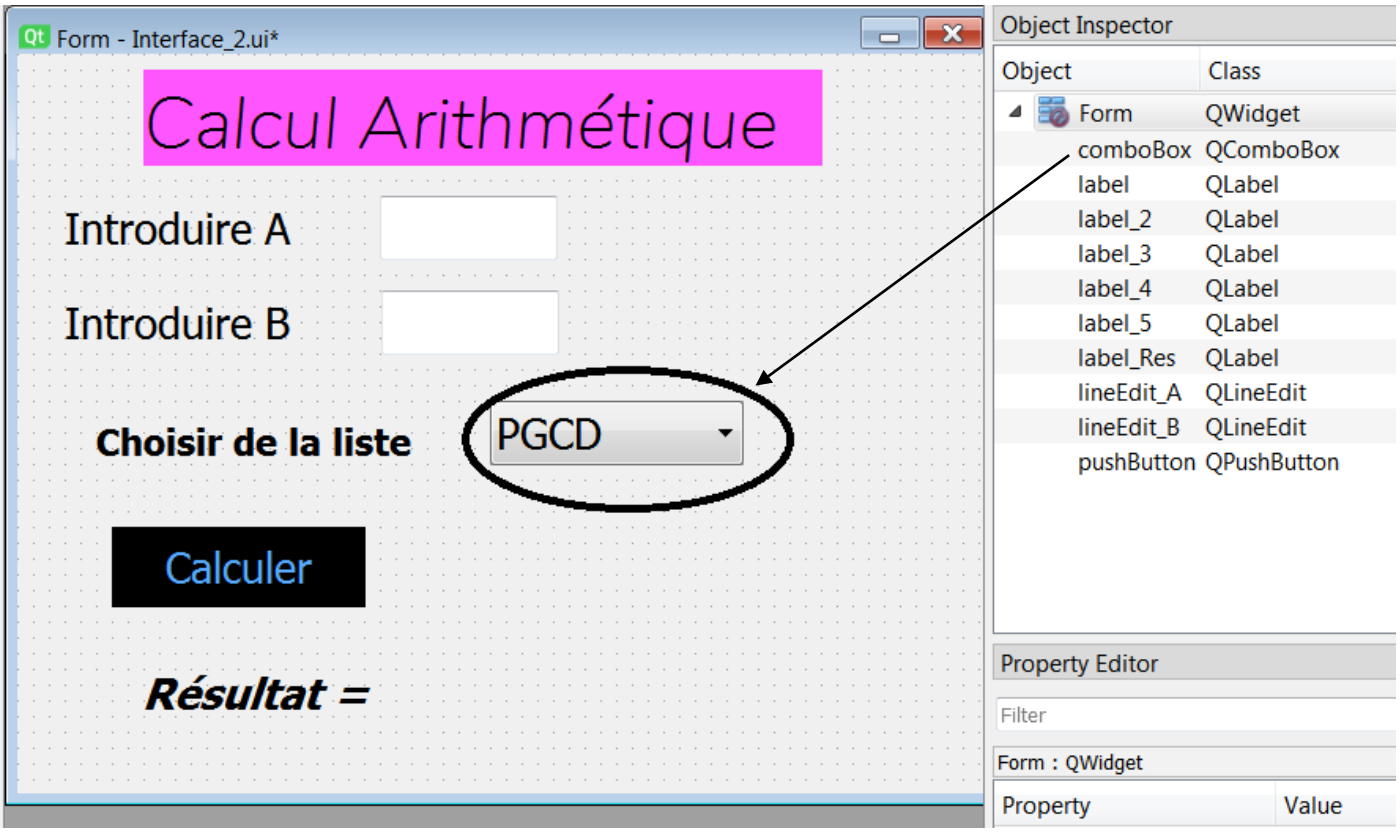
Bouton radio : « QRadioButton » :

Objet (Widget)	Méthode	Rôle
Radio Button fenetre.nom_radiobutton.	isChecked ( )	retourne True si le bouton Radio est coché et False dans le cas contraire
	setChecked (True/ False)	permet de modifier le cochage d'un bouton radio à True (coché) ou à False (non coché)
	text ( )	Une méthode qui permet de récupérer le texte existant dans un objet «Radio Button».
	setText (val)	une méthode qui permet de modifier le texte (la valeur) de l'objet «Radio Button»

Activité : Voir l'exécution du programme Arithmétique-2.py

Application 2 :

1- Créer l'interface suivante et l'enregistrer dans le dossier D:\APPLICATION_2 sous le nom Interface_2.UI



2- Cliquer sur le bouton « **Ajouter code PyQt5** » qui permet d'insérer un script Python permettant d'appeler le fichier (**D:APPLICATION_2)**« ***Interface-2.ui*** », l'enregistrer sous le nom «***arithmétique-2.py*** »

3- Compléter le programme et l'exécuter

Liste déroulante : « QComboBox »

Objet (Widget)	Méthode	Rôle
Combo Box fenetre.nom_combobox.	currentIndex ( )	Une méthode qui retourne l'indice de l'élément sélectionné
	addItem (val)	Une méthode qui permet d'ajouter une valeur
	count ( )	Une méthode qui retourne le nombre d'éléments dans une liste déroulante
	currentText ( )	Une méthode qui permet de récupérer la valeur sélectionnée dans un objet «Combo Box».
	itemText (ind)	Une method qui retourne l'élément de la liste déroulante qui existe dans l'indice «ind»

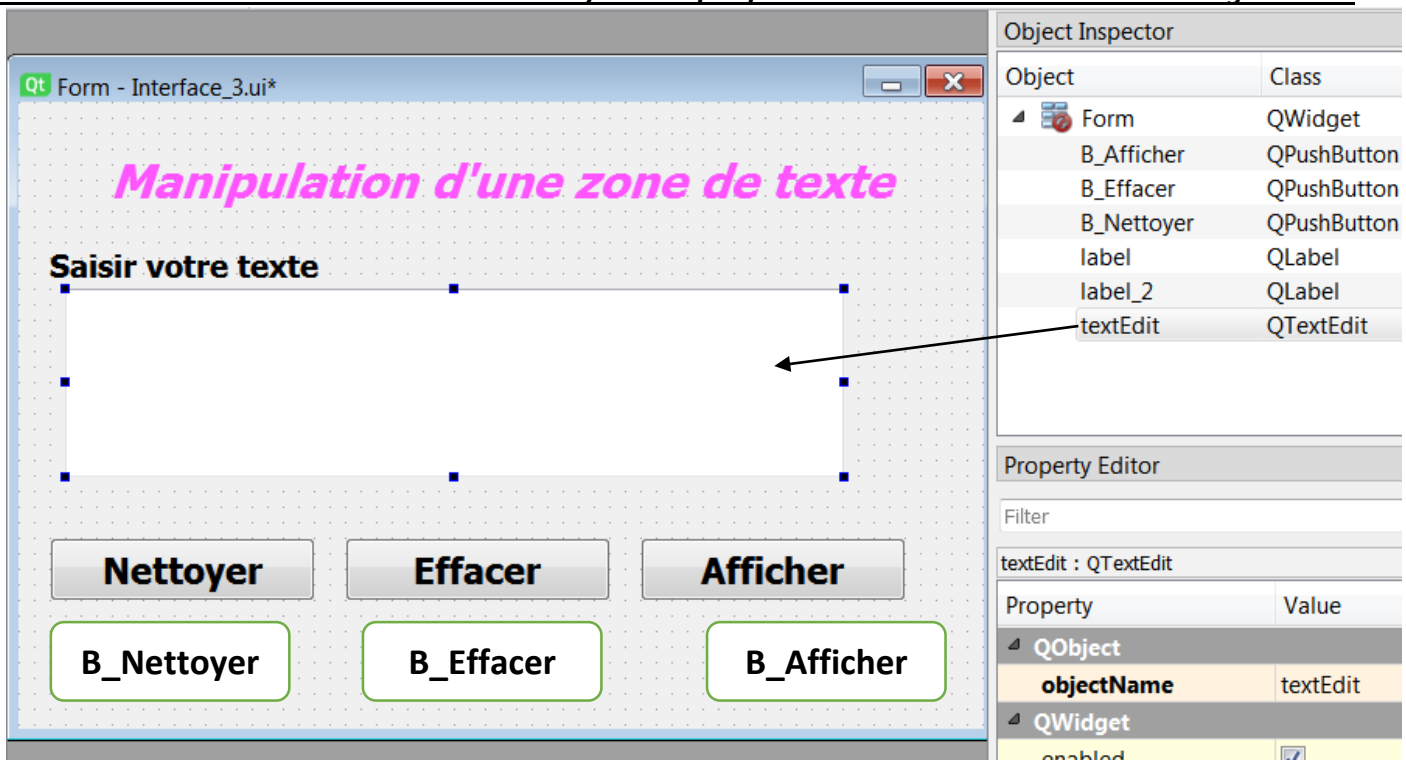
```
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
def pgcd(a,b):
    while (a!=b):
        if a>b:
            a=a-b
        else:
            b=b-a
    return a
def ppcm(a,b):
    return a*b/pgcd(a,b)
def puissance(a,b):
    p=1
    for i in range(b):
        p=p*a
    return p
def pushButton_click():

    a=int(fenetre.lineEdit_A.text())
    b=int(fenetre.lineEdit_B.text())
    if fenetre.comboBox.currentIndex()==0 :
        r=pgcd(a,b)
    elif fenetre.comboBox.currentIndex()==1 :
        r=ppcm(a,b)
    else :
        r=puissance(a,b)
    fenetre.label_Res.setText(str(r))
app = QApplication([])
fenetre = loadUi ("D:/APPLICATION_QT/APPLICATION_2/Interface_2.ui")
fenetre.show()
fenetre.pushButton.clicked.connect ( pushButton_click )
app.exec_()
```

Activité : Voir l'exécution du programme Application_3.py

Application 3 :

1- Créer une interface graphique comme le montre la figure suivante :



2- Enregistrer cette interface sous le nom Interface_3.ui dans le dossier D:\APPLICATION_3

3- Créer un nouveau fichier python puis Cliquer sur le bouton « **Ajouter code PyQt5** » qui permet d'insérer un script Python permettant d'appeler le fichier (**D:APPLICATION_3)**« ***Interface-3.ui*** », l'enregistrer sous le nom «***APPLICATION_3.py*** »

4- Compléter le programme en suivant les indications suivantes :

Lorsqu'on lance le programme :

i. l'utilisateur tape un texte de plusieurs lignes dans la zone texte
ii. Lorsqu'on clique sur le bouton Nettoyer il y a appel d'un module qui efface les espaces superflues de texte
iii. Lorsqu'on clique sur le bouton la zone saisie sera vidée
iv. Lorsqu'on on clique sur le bouton Afficher il y affichage de la chaine nettoyée