

# Chapitre 3
# Gestion des styles et apparences en PyQt6

## 1. Introduction

PyQt6 apporte quelques changements par rapport à PyQt5, notamment au niveau de certains modules, méthodes, et classes. Cependant, les principes de gestion des styles restent similaires, offrant la flexibilité de personnaliser l'apparence des applications.

Dans ce chapitre, nous approfondirons l'utilisation des polices de caractères, des images et des icônes pour enrichir visuellement nos applications PyQt. Nous explorerons plus largement la gestion des **ressources** dans PyQt. Nous détaillerons également l'utilisation de `QPalette`, `QBrush` et des **feuilles de style** en PyQt, en découvrant notamment les *Qt Style Sheets* (QSS). Enfin, nous apprendrons à créer nos propres styles personnalisés pour PyQt.

Pour assimiler progressivement les concepts abordés, nous développerons une application qui servira de fil conducteur tout au long du chapitre. Il s'agit d'un formulaire très simple permettant de saisir un nom, un prénom (dans des zones de texte), un loisir préféré (dans une *combobox*), ainsi que la possession (ou non) d'un vélo (précision à l'aide d'une *checkbox*).

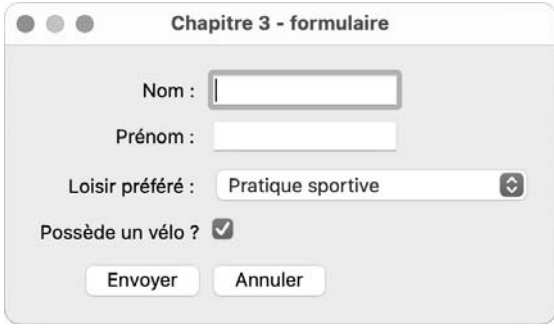
Le code de cette application est le suivant :

```
import sys
from PyQt6.QtWidgets import (
   QApplication,
```

```
    QWidget,
    QPushButton,
    QCheckBox,
    QFormLayout,
    QLabel,
    QLineEdit,
    QComboBox
)
from PyQt6.QtCore import Qt

class Fenetre(QWidget):
    def __init__(self):
        super().__init__()

        self.setGeometry(100, 100, 250, 100)
        self.setWindowTitle("Chapitre 3 - formulaire")

        self.disposition = QFormLayout()

        # Champ "Nom"
        self.nomLabel = QLabel("Nom : ")
        self.nom = QLineEdit()
        self.disposition.addRow(self.nomLabel, self.nom)

        # Champ "Prénom"
        self.prenomLabel = QLabel("Prénom : ")
        self.prenom = QLineEdit()
        self.disposition.addRow(self.prenomLabel, self.prenom)

        # Champ "Loisir préféré"
        self.loisirLabel = QLabel("Loisir préféré : ")
        self.loisir = QComboBox()
        self.loisir.addItems([
            "Pratique sportive",
            "Pratique artistique",
            "Programmation informatique",
            "Voyage"
        ])
        self.disposition.addRow(self.loisirLabel, self.loisir)

        # Checkbox "Possède un vélo ?"
        self.veloLabel = QLabel("Possède un vélo ?")
        self.velo = QCheckBox()
        self.velo.setChecked(True)
```

```
        self.disposition.addRow(self.veloLabel, self.velo)

        # Boutons "Envoyer" et "Annuler"
        self.boutonEnvoyer = QPushButton("Envoyer")
        self.boutonAnnuler = QPushButton("Annuler")
        self.disposition.addRow(self.boutonEnvoyer,
self.boutonAnnuler)

        # Définir la disposition
        self.setLayout(self.disposition)

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

Graphiquement cette petite application ressemble à ceci :




*Fenêtre de formulaire*

À titre informatif voici le code équivalent en **PyQt5** ; si les différences sont effectives, on peut se rendre compte que les différences restent ténues et se limitent à quelques aspects déclaratifs.

```
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton,
QCheckBox from PyQt5.QtWidgets import QFormLayout, QLabel,
QLineEdit, QComboBox


class Fenetre(QWidget): def__init__(self):
```

```
super().__ init__ ()

self.setGeometry(100, 100, 250, 100)
self.setWindowTitle("Chapitre 3 - formulaire")
self.disposition = QFormLayout()

self.nomLabel = QLabel("Nom : ") self.nom = QLineEdit()
self.disposition.addRow(self.nomLabel, self.nom)

self.prenomLabel = QLabel("Prénom : ") self.prenom = QLineEdit()
self.disposition.addRow(self.prenomLabel, self.prenom)

self.loisirLabel = QLabel("Loisir préféré : ")
self.loisir = QComboBox()

self.loisir.addItems(["Pratique sportive", "Pratique artistique",
"Programmation informatique", "Voyage"])
self.disposition.addRow(self.loisirLabel, self.loisir)

self.veloLabel = QLabel("Possède un vélo ?") self.velo =
QCheckBox() self.velo.setChecked(True)
self.disposition.addRow(self.veloLabel, self.velo)

self.boutonEnvoyer = QPushButton("Envoyer")
self.boutonAnnuler = QPushButton("Annuler")


self.disposition.addRow(self.boutonEnvoyer, self.boutonAnnuler)
self.setLayout(self.disposition)
```

Entrons dans le vif du sujet, en commençant par la gestion des polices et donc par `QFont`.

# 2. Manipulation des polices et l'objet QFont

## 2.1 Utilisation de QFont

L'objet `QFont` permet d'encapsuler le nom de la police choisie, sa taille, son style éventuel (gras, italique, etc.), son étirement, la dimension des espaces entre chaque caractère et bien d'autres paramètres encore.

Pour l'utiliser, on commence par déclarer l'import adéquat.

```
from PyQt6.QtGui import QFont
```

On instancie ensuite l'objet de type `QFont`, par exemple ci-dessous dans lequel on déclare la police de nom **Chalkduster**, en taille **18** et avec une stylisation en **gras**. On active également la mise en **italique**.

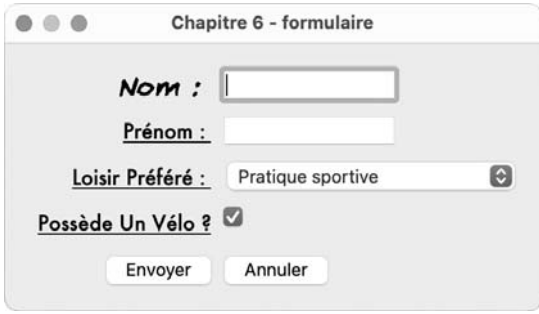
```
maFont = QFont('Chalkduster', 18, QFont.Weight.Bold)
maFont.setItalic(True)
```

Une fois notre police PyQt créée, on peut l'utiliser sur nos différents widgets grâce à la méthode `setFont`. Ainsi, en reprenant l'exemple précédent, on change la police de différents labels de l'interface.

```
import sys
from PyQt6.QtWidgets import (
   QApplication, QWidget, QPushButton, QCheckBox,
   QFormLayout, QLabel, QLineEdit, QComboBox
)
from PyQt6.QtGui import QFont
from PyQt6.QtCore import Qt


# Première police
maFont = QFont('Chalkduster', 18, QFont.Weight.Bold)
maFont.setItalic(True)

# Deuxième police
maFont2 = QFont()
maFont2.setFamily('Futura')
maFont2.setPointSize(15)
maFont2.setCapitalization(QFont.Capitalization.Capitalize)
maFont2.setWeight(QFont.Weight.Medium)
```

```
maFont2.setUnderline(True)


class Fenetre(QWidget):
   def __init__(self):
       super().__init__()

       self.setGeometry(100, 100, 250, 100)
       self.setWindowTitle("Chapitre 3 - formulaire")
       self.disposition = QFormLayout()

       self.nomLabel = QLabel("Nom : ")
       self.nom = QLineEdit()
       self.disposition.addRow(self.nomLabel, self.nom)

       self.prenomLabel = QLabel("Prénom : ")
       self.prenom = QLineEdit()
       self.disposition.addRow(self.prenomLabel, self.prenom)

       self.loisirLabel = QLabel("Loisir préféré : ")
       self.loisir = QComboBox()
       self.loisir.addItems([
           "Pratique sportive", "Pratique artistique",
           "Programmation informatique", "Voyage"
       ])
       self.disposition.addRow(self.loisirLabel, self.loisir)

       self.veloLabel = QLabel("Possède un vélo ?")
       self.velo = QCheckBox()
       self.velo.setChecked(True)
       self.disposition.addRow(self.veloLabel, self.velo)

       self.boutonEnvoyer = QPushButton("Envoyer")
       self.boutonAnnuler = QPushButton("Annuler")
       self.disposition.addRow(self.boutonEnvoyer,
self.boutonAnnuler)

       self.setLayout(self.disposition)

       # Appliquer les polices
       self.nomLabel.setFont(maFont)
       self.prenomLabel.setFont(maFont2)
       self.loisirLabel.setFont(maFont2)
       self.veloLabel.setFont(maFont2)
```

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




*Rendu de l'exemple d'usage de QFont*

## 2.2 La question des polices de caractères non installées

Les polices de caractères utilisées dans vos documents ou définies programmatiquement via PyQt sont généralement préinstallées sur l'ordinateur cible. Elles proviennent soit du système d'exploitation, soit d'une installation manuelle.

Cependant, il est difficile de garantir qu'une police donnée sera disponible sur toutes les machines où votre application PyQt sera exécutée. Il est donc recommandé d'utiliser des polices courantes et d'anticiper l'absence d'une police spécifique sur la machine cible.

Lorsque la police demandée n'est pas disponible, PyQt utilise une police par défaut. Pour comprendre ce comportement et identifier la police effectivement utilisée, on peut utiliser la classe `QFontInfo`.

### Exemple pratique avec une police inexistante

On commence par créer un objet `QFont` avec une famille de polices fictive, par exemple `"LA_POLICE_INCONNUE"`, qui n'existe sur aucune machine. Voici comment définir cette police :

```
maFont = QFont()
maFont.setFamily('LA_POLICE_INCONNUE')
maFont.setPointSize(15)
maFont.setCapitalization(QFont.Capitalization.Capitalize)
maFont.setWeight(QFont.Weight.Medium)
```

Lorsque la police n'est pas trouvée, **PyQt** remplace automatiquement cette dernière par une police par défaut. Le code complet pour tester ce comportement est le suivant.

```
import sys
from PyQt6.QtWidgets import (
   QApplication, QWidget, QPushButton, QCheckBox,
   QFormLayout, QLabel, QLineEdit, QComboBox
)
from PyQt6.QtGui import QFont, QFontInfo

# Création de la police inexistante
maFont = QFont()
maFont.setFamily('LA_POLICE_INCONNUE')  # Police inexistante
maFont.setPointSize(15)
maFont.setCapitalization(QFont.Capitalization.Capitalize)
maFont.setWeight(QFont.Weight.Medium)

class Fenetre(QWidget):
   def __init__(self):
       super().__init__()

       self.setGeometry(100, 100, 350, 150)
       self.setWindowTitle("Chapitre 3 - QFontInfo")
       self.disposition = QFormLayout()

       # Création des champs du formulaire
       self.nomLabel = QLabel("Nom : ")
       self.nom = QLineEdit()
       self.nomLabel.setFont(maFont)
# Application de la police inexistante
       self.disposition.addRow(self.nomLabel, self.nom)
```

```
        self.prenomLabel = QLabel("Prénom : ")
        self.prenom = QLineEdit()
        self.disposition.addRow(self.prenomLabel, self.prenom)

        self.loisirLabel = QLabel("Loisir préféré : ")
        self.loisir = QComboBox()
        self.loisir.addItems([
            "Pratique sportive", "Pratique artistique",
            "Programmation informatique", "Voyage"
        ])
        self.disposition.addRow(self.loisirLabel, self.loisir)

        self.veloLabel = QLabel("Possède un vélo ?")
        self.velo = QCheckBox()
        self.velo.setChecked(True)
        self.disposition.addRow(self.veloLabel, self.velo)

        # Boutons
        self.boutonEnvoyer = QPushButton("Envoyer")
        self.boutonAnnuler = QPushButton("Annuler")
        self.disposition.addRow(self.boutonEnvoyer,
self.boutonAnnuler)

        self.setLayout(self.disposition)

        # Identification de la police effectivement utilisée
        testFont = QFontInfo(maFont)
        print('Police utilisée par défaut :', testFont.family())

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

Quand on exécute ce programme dans le terminal, ici sur une machine macOS, on obtient le retour suivant (en plus de l'affichage de la fenêtre utilisant la police par défaut, ici `.AppleSystemUIFont`).

```
> python3 QFontInfo_Police_non_existante.py
qt.qpa.fonts: Populating font family aliases took 48 ms. Replace
uses of missing font family "LA_POLICE_INCONNUE" with one that
exists to avoid this cost.
```