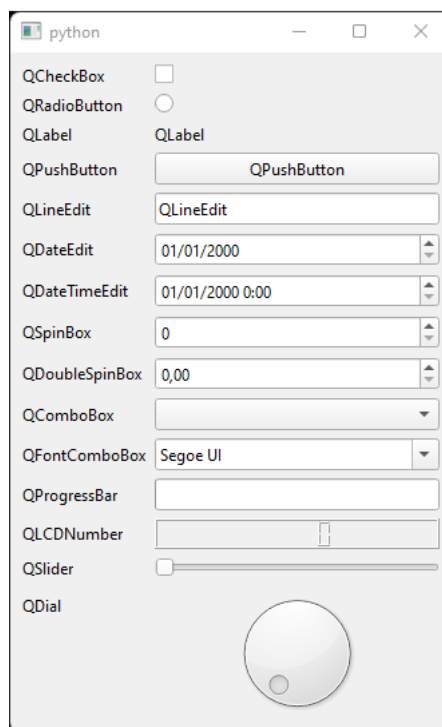


TEMATITZACIÓ

Contenido

1. Estils d'aplicació.....	1
2. Paleta de colors	3
3. Utilització d'icones	4
4. QSS: Qt Style Sheets	5
5. Carrega de fitxers QSS	7



1. Estils d'aplicació

Els estils modifiquen l'estètica dels components. Per defecte, Qt aplica estils específics per a cada plataforma per integrar l'aplicació visualment, aquesta és la raó per la qual el mateix programa es veurà diferent a Windows, Linux i Mac.

Els estils es poden personalitzar per no fer-los dependents de la plataforma i de fet el propi Qt té un tema anomenat Fusion que proveeix una estètica multiplataforma i moderna. He preparat un formulari amb tot tipus de widgets perquè puguem apreciar els canvis visuals en canviar els estils:

```
from PySide6.QtWidgets import (  
    QApplication, QMainWindow, QFormLayout, QWidget, QLineEdit,  
    QSpinBox)  
from PySide6.QtCore import Qt
```

```

import sys

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        formulario = QFormLayout()

        formulario.addRow("Nombre", QLineEdit("Hector"))
        formulario.addRow("Email", QLineEdit(text="hola@ejemplo.com"))
        formulario.addRow("Edad", QSpinBox(value=32))

        widget = QWidget()
        widget.setLayout(formulario)

        self.setCentralWidget(widget)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec_())

```

Per activar el tema Fusion només cal trucar al mètode setStyle de l'aplicació:

```

if __name__ == "__main__":
    app = QApplication(sys.argv)

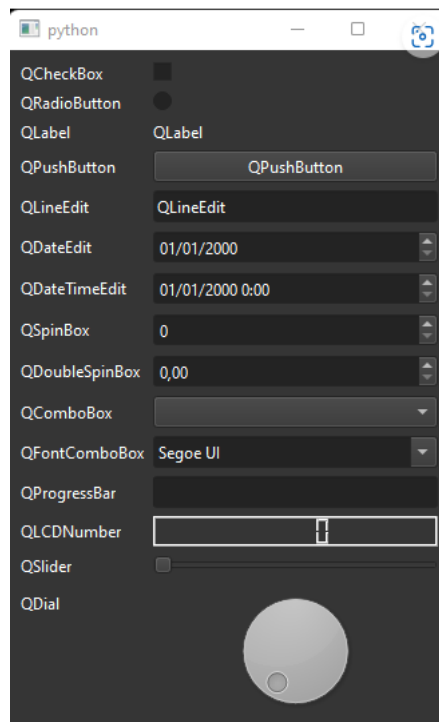
    # estilo fusion
    app.setStyle("Fusion")

    window = MainWindow()
    window.show()
    sys.exit(app.exec_())

```

Com podreu observar són canvis molt subtils que almenys a Windows costa apreciar a simple vista, però així ens assegurem de visualitzar la mateixa estètica a totes les plataformes.

2. Paleta de colors



La selecció de colors que utilitza Qt per dibuixar els components es fa servir en paletes. Anem a experimentar amb la paleta de colors de l'aplicació de la lliçó anterior:

```
from PySide6.QtGui import QPalette, QColor # nuevo

if __name__ == "__main__":
    app = QApplication(sys.argv)

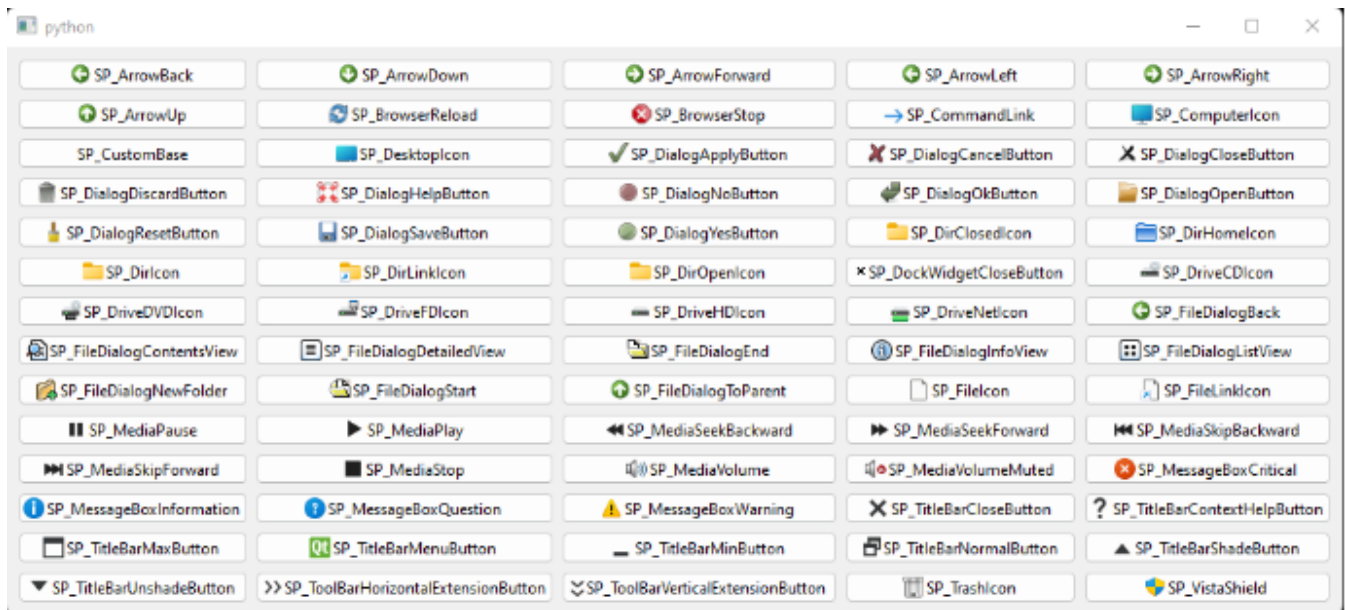
    # creamos nuestra paleta de colores
    paleta = QPalette()
    paleta.setColor(QPalette.Window, QColor(51, 51, 51))
    paleta.setColor(QPalette.WindowText, QColor(235, 235, 235))

    # activamos la paleta en la aplicación
    app.setPalette(paleta)

    window = MainWindow()
    window.show()
    sys.exit(app.exec_())
```

Com veieu les paletes tenen accessors per establir els colors dels diferents components. [Més info.](#)

3. Utilització d'icones



Anteriorment ja hem utilitzat algunes icones carregant-les com a recursos externs, però Qt inclou un set d'icones predeterminades. Podem fer-ne ús de la manera següent:

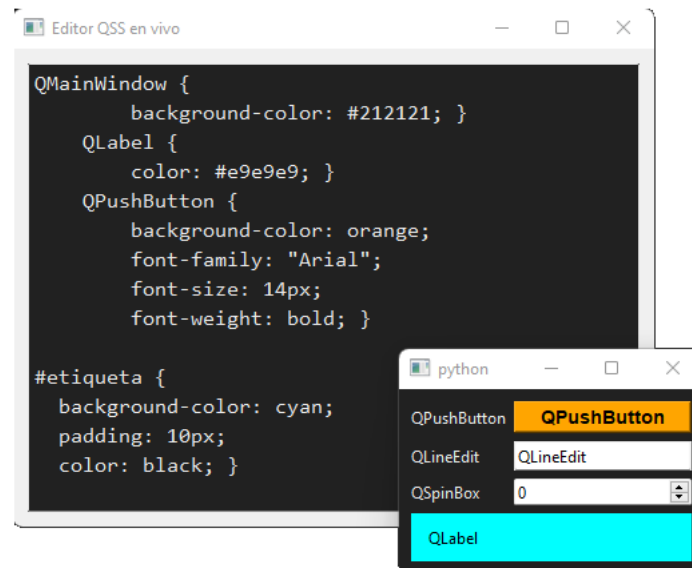
```
from PySide6.QtWidgets import (
    QApplication, QMainWindow, QWidget, QPushButton, QStyle) # edited
import sys

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        # recuperamos el icono de la librería estandar de la ventana
        icono = self.style().standardIcon(QStyle.SP_DialogSaveButton)
        # lo podemos asignar a un botón
        boton = QPushButton(icono, "Botón guardar")

        self.setCentralWidget(boton)
```

4. QSS: Qt Style Sheets



L'última cosa que veurem sobre tematzació són els fulls d'estil de Qt, o abreujats QSS. Si sabeu una mica de programació web segur que us sona el llenguatge CSS, ja que QSS és una forma d'afegir estil als widgets utilitzant pràcticament la mateixa sintaxi.

En aquesta pràctica personalitzarem uns quants widgets bàsics dins d'un layout estil formulari:

```
from PySide6.QtWidgets import (
    QApplication, QMainWindow, QFormLayout, QWidget, QLabel,
    QRadioButton,
    QCheckBox, QLineEdit, QSpinBox, QPushButton, QPlainTextEdit)
from pathlib import Path
import sys

def absPath(file):
    return str(Path(__file__).parent.absolute() / file)

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        formulario = QFormLayout()
        formulario.addRow("QCheckBox", QCheckBox())
        formulario.addRow("QRadioButton", QRadioButton())
        formulario.addRow("QLabel", QLabel("QLabel"))
        formulario.addRow("QPushButton", QPushButton("QPushButton"))
        formulario.addRow("QLineEdit", QLineEdit("QLineEdit"))
        formulario.addRow("QSpinBox", QSpinBox())

        widget = QWidget()
        widget.setLayout(formulario)
        self.setCentralWidget(widget)
```

La forma més senzilla d'establir els estils és a través del mètode `setStyleSheet` del widget principal, ja que amb ell podem donar estil a tot allò que conté:

```
# estils QSS
self.setStyleSheet("""
    QMainWindow {
        background-color: #212121; }
    QLabel {
        color: #e9e9e9; }
    QPushButton {
        background-color: orange;
        font-family: "Arial";
        font-size: 14px;
        font-weight: bold; }
""")
```

Ara bé, aquests estils són globals i afecten totes les instàncies. Si volem estilitzar una sola instància podem atorgar-li un identificador:

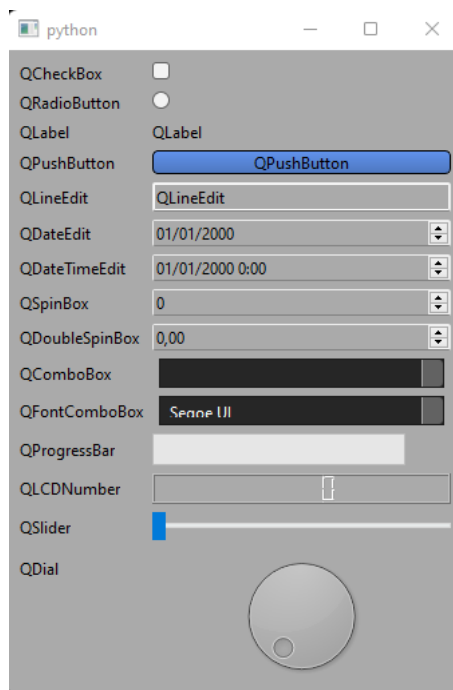
```
etiqueta = QLabel("QLabel")
etiqueta.setObjectName("etiqueta")
formulario.addRow(etiqueta)
```

I referir-nos a ella en QSS usant la coixinet (almohadilla) igual que en CSS:

```
"""
#etiqueta {
    background-color: cyan;
    padding: 10px;
    color: black; }
"""
```

La veritat és que aquest tema abasta molt i no em vull estendre, us deixaré la [documentació](#) amb totes les propietats disponibles i cadascú que aprofundeixi en la mesura del necessari.

5. Carrega de fitxers QSS



En aquesta darrera lliçó carregarem fitxers QSS per no haver d'escriure el codi al mateix programa. He preparat un bon grapat d'estils que he trobat per Internet, us els adjunte als recursos, sentiu-vos lliures d'utilitzar-los respectant les directrius de cada creador, la font del qual trobareu a la capçalera de cada fitxer. Tinc un programa ja preparat per començar a treballar:

```
from PySide6.QtWidgets import (
    QApplication, QMainWindow, QFormLayout, QWidget, QLabel,
    QRadioButton, QCheckBox, QLineEdit, QSpinBox, QDoubleSpinBox,
    QPushButton, QComboBox, QFontComboBox, QDateEdit, QDateTimeEdit,
    QLCDNumber, QProgressBar, QDial, QSlider)
from PySide6.QtCore import Qt
from pathlib import Path
import sys

def absPath(file):
    return str(Path(__file__).parent.absolute() / file)

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        formulario = QFormLayout()

        formulario.addRow("QCheckBox", QCheckBox())
        formulario.addRow("QRadioButton", QRadioButton())
        formulario.addRow("QLabel", QLabel("QLabel"))
```

```

formulario.addRow("QPushButton", QPushButton("QPushButton"))
formulario.addRow("QLineEdit", QLineEdit("QLineEdit"))
formulario.addRow("QDateEdit", QDateEdit())
formulario.addRow("QDateTimeEdit", QDateTimeEdit())
formulario.addRow("QSpinBox", QSpinBox())
formulario.addRow("QDoubleSpinBox", QDoubleSpinBox())
formulario.addRow("QComboBox", QComboBox())
formulario.addRow("QFontComboBox", QFontComboBox())
formulario.addRow("QProgressBar", QProgressBar())
formulario.addRow("QLCDNumber", QLCDNumber())
formulario.addRow("QSlider", QSlider(Qt.Horizontal))
formulario.addRow("QDial", QDial())

widget = QWidget()
widget.setLayout(formulario)

self.setCentralWidget(widget)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec_())

```

Per carregar els estils m'ajudaré de la funció `absPath`, obriré els fitxers d'estil com si fossin text, llegiré el contingut i el bolcaré al mètode `setStyleSheet` de la finestra:

```

def cargarQSS(self, file):
    # guardamos la ruta absoluta al fichero
    path = absPath(file)
    # intentamos abrirlo y volcar el contenido
    try:
        with open(path) as styles:
            self.setStyleSheet(styles.read())
    # si hay algún fallo lo capturamos con una excepción genérica
    except:
        print("Error abriendo estilos", path)

```

Només resta trucar al mètode i provar alguns temes:

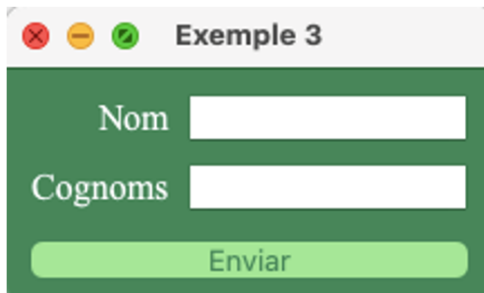
```

# cargamos los estilos del fichero
self.cargarQSS("qss/Ubuntu.qss")
self.cargarQSS("qss/ElegantDark.qss")
self.cargarQSS("qss/ChatBee.qss")
self.cargarQSS("qss/EasyCode.qss")

```


EXEMPLES:

Fes un programa que mostre el següent formulari amb els estils especificats:



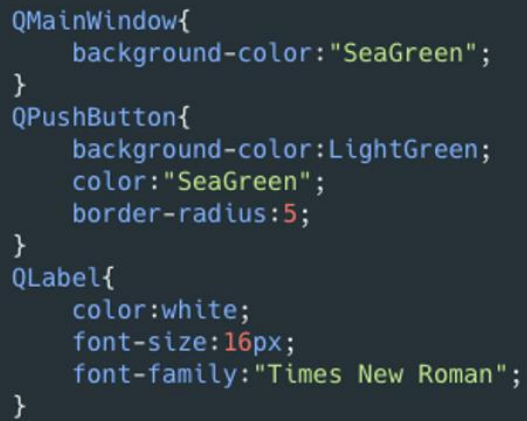
Hem d'aplicar els següents estils:

- Color de fons de la finestra: SeaGreen.
- Color de fons del botó: LightGreen.
- Color del text del botó: SeaGreen.
- Bordejat del botó: 5
- Les etiquetes seran de color blanc, lletra de 16px i de tipus Times New Roman.

Farem ús del QSS: Qt Style Sheets

SOLUCIÓ

```
1 from PySide6.QtWidgets import *
2 from pathlib import Path
3
4 def absPath(file):
5     return str(Path(__file__).parent.absolute() / file)
6
7 class Finestra(QMainWindow):
8     def __init__(self):
9         super().__init__()
10        self.setWindowTitle("Exemple 3")
11        formulari=QFormLayout()
12        formulari.addRow(QLabel("Nom"),QLineEdit())
13        formulari.addRow(QLabel("Cognoms"),QLineEdit())
14        formulari.addRow(QPushButton("Enviar"))
15        contenidor=QWidget()
16        contenidor.setLayout(formulari)
17        self.setCentralWidget(contenidor)
18        self.cargarEstilos("estils.qss")
19
20    def cargarEstilos(self, file):
21        # guardamos la ruta absoluta al fichero
22        path = absPath(file)
23        # intentamos abrirlo y volcar el contenido
24        try:
25            with open(path) as styles:
26                self.setStyleSheet(styles.read())
27            # si hay algún fallo lo capturamos con una excepción genérica
28        except:
29            print("Error abriendo estilos", path)
30
31 if __name__ == "__main__":
32     app=QApplication([])
33     finestra=Finestra()
34     finestra.show()
35     app.exec()
```



```
QMainWindow{
    background-color:"SeaGreen";
}
QPushButton{
    background-color:LightGreen;
    color:"SeaGreen";
    border-radius:5;
}
QLabel{
    color:white;
    font-size:16px;
    font-family:"Times New Roman";
}
```

estils.qss