Fifth Part

انواع دیالوگ‌های آماده و سفارشی PyQt6

۱. QDialog — پایه‌ی دیالوگ‌ها

خب **QDialog** نمایانگر یک پنجره‌ی فرعی (dialog) است که می‌تواند modal (مسدودکننده‌ی پنجره‌ی والد) یا modeless باشد TutorialsPoint.

۱.۱ ایجاد یک QDialog ساده

Copy Python

```
from PyQt6.QtWidgets import QApplication, QWidget, QDialog, QPushButton
from PyQt6.QtCore import Qt
import sys

app = QApplication(sys.argv)
main_win = QWidget()
main_win.setWindowTitle('Main Window')
main_win.resize(300, 200)

# create and configure a dialog
dlg = QDialog(main_win)                    # parent = main window
dlg.setWindowTitle('Simple Dialog')
dlg.setWindowModality(Qt.ApplicationModal)  # make it modal :contentRefer
dlg.resize(200, 100)

btn_open = QPushButton('Open Dialog', main_win)
btn_open.move(100, 80)

def open_dialog():
    dlg.exec()  # show modal dialog :contentReference[oaicite:2]{index=2}

btn_open.clicked.connect(open_dialog)
```

```
main_win.show()
sys.exit(app.exec())
```

- `exec()` پنجره را به صورت modal نمایش می‌دهد و تا بسته نشدن دیالوگ، به پنجره‌ی والد بازنمی‌گردد TutorialsPoint.
- برای نمایش modeless از `dlg.show()` استفاده می‌شود.

۲. QMessageBox — دیالوگ پیام

باید بدانیم **QMessageBox** برای نمایش پیام‌های اطلاعاتی، اخطار یا سوال به کاربر است و شامل آیکن و دکمه‌های استاندارد می‌شود TutorialsPoint.

۲.۱ مثال پیام اطلاعاتی

```
from PyQt6.QtWidgets import QMessageBox

def show_info(parent):
    msg = QMessageBox(parent)
    msg.setIcon(QMessageBox.Information)             # information icon
    msg.setWindowTitle('Information')
    msg.setText('عملیات با موفقیت انجام شد')
    msg.setStandardButtons(QMessageBox.Ok)           # only OK button
    msg.exec()                                       # modal :contentRef
```

۲.۲ سوال با چند گزینه

```
def ask_question(parent):
    msg = QMessageBox(parent)
    msg.setIcon(QMessageBox.Question)
    msg.setWindowTitle('Confirm')
    msg.setText('آیا از ادامه‌ی عملیات مطمئن هستید؟')
    msg.setStandardButtons(QMessageBox.Yes | QMessageBox.No)  # Yes/No :c
    result = msg.exec()
    if result == QMessageBox.Yes:
```

```
        print('User chose Yes')
    else:
        print('User chose No')
```

۳. QFileDialog — انتخاب فایل

کاربرد **QFileDialog** دیالوگی برای انتخاب فایل یا پوشه است. هم متدهای ایستا (`getOpenFileName`) و هم شیء (`QFileDialog`) پشتیبانی می‌شود TutorialsPoint.

۳.۱ انتخاب یک فایل برای باز کردن

```
from PyQt6.QtWidgets import QFileDialog

def open_file(parent):
    fname, _ = QFileDialog.getOpenFileName(
        parent,
        'Open file',             # dialog title
        '',                       # start directory
        'Text files (*.txt);;All files (*)'  # filter
    )  # static method :contentReference[oaicite:9]{index=9}
    if fname:
        print('Selected file:', fname)
```

۳.۲ انتخاب چند فایل و ذخیره

```
def save_file(parent):
    fname, _ = QFileDialog.getSaveFileName(
        parent,
        'Save file',
        'untitled.txt',
        'Text files (*.txt)'
    )
    if fname:
        print('Save to:', fname)
```

۴. دیالوگ سفارشی با ارث‌بری از QDialog

گاهی نیاز به دیالوگ با فیلدهای اختصاصی داریم. با subclassing می‌توان فرم دلخواه ساخت Stack Overflow.

```
from PyQt6.QtWidgets import QDialog, QLineEdit, QPushButton, QVBoxLayout

class CustomDialog(QDialog):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.setWindowTitle('Custom Dialog')
        # input field
        self.edit = QLineEdit(self)
        self.edit.setPlaceholderText('Enter your name')
        # OK button
        self.btn = QPushButton('OK', self)
        self.btn.clicked.connect(self.accept)  # built-in slot to close w
        # layout
        layout = QVBoxLayout(self)
        layout.addWidget(self.edit)
        layout.addWidget(self.btn)

    def get_input(self):
        return self.edit.text()

# usage in main window
def open_custom(parent):
    dlg = CustomDialog(parent)
    if dlg.exec() == QDialog.Accepted:
        name = dlg.get_input()
        print('Name entered:', name)
```

- متد `accept()` دیالوگ را با کد `Accepted` می‌بندد.
- می‌توان اسلات‌های سفارشی نیز تعریف کرد.

۵. ترکیب دیالوگ‌ها در یک اپلیکیشن

مثال: پنجره‌ای با سه دکمه برای نمایش دیالوگ ساده، پیام و دیالوگ سفارشی

```
import sys
from PyQt6.QtWidgets import QApplication, QWidget, QPushButton, QVBoxLayc

class MainWin(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle('Dialogs Demo')
        layout = QVBoxLayout(self)

        btn1 = QPushButton('Show Simple Dialog', self)
        btn1.clicked.connect(self.show_simple_dialog)

        btn2 = QPushButton('Show Message', self)
        btn2.clicked.connect(self.show_info)

        btn3 = QPushButton('Custom Dialog', self)
        btn3.clicked.connect(self.open_custom)

        layout.addWidget(btn1)
        layout.addWidget(btn2)
        layout.addWidget(btn3)

    def show_simple_dialog(self):
        dlg = QDialog(self)
        dlg.setWindowTitle('Simple Dialog')
        dlg.exec()

    def show_info(self):
        QMessageBox.information(self, 'Info', 'This is a message box.')

    def open_custom(self):
        dlg = QDialog(self)
        dlg.setWindowTitle('Custom Dialog')
        dlg.exec()

app = QApplication(sys.argv)
w = MainWin()
w.show()
```

```
sys.exit(app.exec())
```

تمرین‌ها

۱. دیالوگ `QInputDialog` را برای ورود عدد یا متن امتحان کن.

۲. در `QFileDialog` فقط پوشه انتخاب کن (`getExistingDirectory`).

۳. یک دیالوگ سفارشی با دو فیلد (`QLineEdit`) بساز و خروجی را در لیست اصلی نمایش بده.