LÀM VIỆC VỚI DATE VÀ TIME

Nội dung

- Hiển thị LCD digits
- Hiển thị đồng hồ hệ thống bằng LCD – like digits
- Hiển thị dữ liệu khi người dùng chọn từ Calendar Widget
- Hiển thị bảng dữ liệu bằng Table Widget

Hiển Thị LCD digits

- Qt Designer cung cấp **LCD Number widget** (thuộc lớp **QLCDNumber**) để hiển thị chữ số với kích thước bất kỳ.
- Widget này hỗ trợ nhiều hệ số: **Dec (thập phân), Hex (thập lục phân), Oct (bát phân), Bin (nhị phân)**. Các phương thức chính gồm: **setMode()** để chọn kiểu số, **display()** để hiển thị giá trị, và **value()** để lấy giá trị đang hiển thị.
- Muốn hiển thị **đồng hồ hệ thống tự động cập nhật**, cần sử dụng **timer**.

Timers

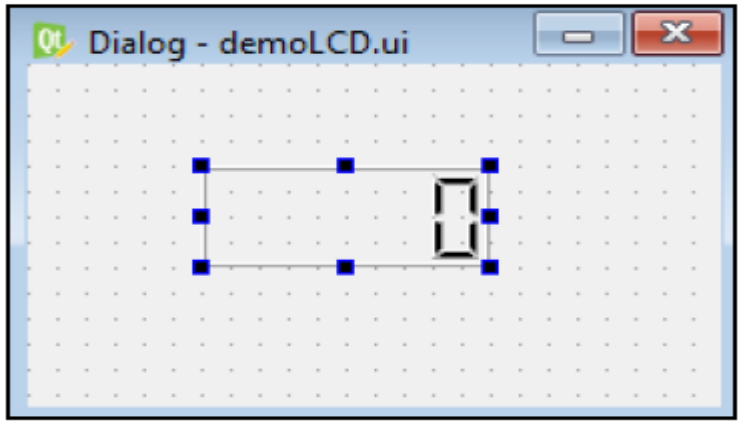
- **QTimer** dùng để thực hiện công việc **lặp lại theo thời gian**.
- Công việc được đặt trong **một phương thức**, phương thức này được gọi khi phát tín hiệu **timeout()**.
- Các cách cấu hình timer:
  - **start(n)**: phát timeout() sau mỗi **n mili giây**.
  - **setSingleShot(true)**: chỉ phát timeout() **một lần**.
  - **singleShot(n)**: phát timeout() **một lần sau n mili giây**.

Qtime class

- **QTime** dùng để lấy và xử lý thời gian hệ thống theo định dạng **24 giờ** (giờ, phút, giây, mili giây).
- Hỗ trợ **đo thời gian trôi qua** và **tính chênh lệch thời gian**.
- Các phương thức chính:
  - currentTime() : lấy thời gian hiện tại
  - hour(), minute(), second(), msec() : lấy từng thành phần thời gian
  - addSecs(), addMsecs() : cộng thêm giây hoặc mili giây
  - secsTo() : tính số giây chênh lệch giữa hai đối tượng QTime

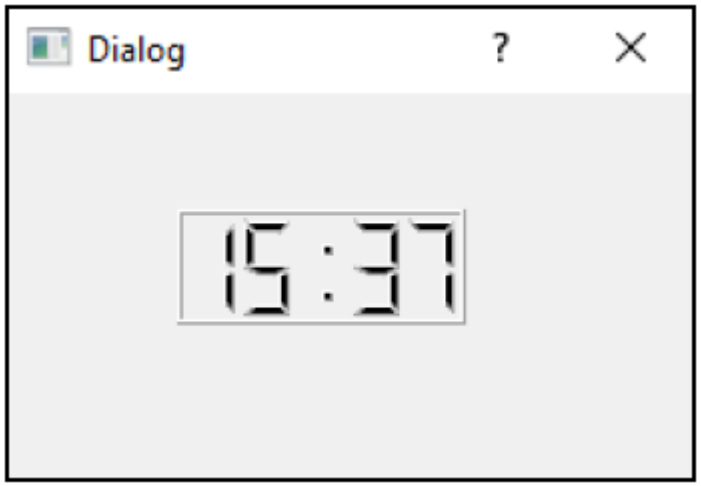
Hiển thị đồng hồ hệ thống bằng LCD – like digits

LCD (Liquid Crystal Display) là loại hiển thị seven-segment, được dùng phổ biến trong các thiết bị điện tử. Trong ví dụ này, ta tạo lớp **QTime** để hiển thị thời gian hiện tại bằng **LCD Number** trong Qt. Các bước chính gồm: tạo giao diện bằng **Qt Designer**, thêm widget LCD Number, thiết lập kích thước, chuyển file .ui sang mã Python bằng **pyuic5**, và viết script Python để gọi và chạy giao diện.



Hiển thị đồng hồ hệ thống bằng LCD – like digits

```
import sys
from PyQt5.QtWidgets import QDialog, QApplication
from demoLCD import *
class MyForm(QDialog):
    def __init__(self):
        super().__init__()
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)
        timer = QtCore.QTimer(self)
        timer.timeout.connect(self.showlcd)
        timer.start(1000)
        self.showlcd()
    def showlcd(self):
        time = QtCore.QTime.currentTime()
        text = time.toString('hh:mm')
        self.ui.lcdNumber.display(text)
if __name__=="__main__":
    app = QApplication(sys.argv)
    w = MyForm()
    w.show()
    sys.exit(app.exec_())
```



Hiển thị dữ liệu khi người dùng chọn từ Calendar Widget

- **Calendar Widget** và **Date Edit** dùng để hiển thị và chọn ngày.

Khi chọn ngày trên Calendar Widget thì Date Edit tự động cập nhật theo.

- **Khác nhau**:

Calendar Widget lớn, trực quan, dễ đọc;
Date Edit gọn hơn nhưng kém trực quan.

- **QDate**: lấy ngày từ đồng hồ hệ thống.

Calendar Widget (QCalendarWidget)

- **Chức năng:** Cho phép người dùng chọn ngày bằng lịch tháng.
- **Mặc định:** Hiển thị tháng/năm hiện tại; ngày ở dạng rút gọn (Mon, Tue…); Thứ Bảy & Chủ Nhật màu đỏ; Chủ Nhật ở cột đầu tiên.

Thuộc tính chính:

- minimumDate, maximumDate: giới hạn khoảng ngày cho phép chọn.
- selectionMode: bật/tắt khả năng chọn ngày (NoSelection = không cho chọn).
- VerticalHeaderFormat: ẩn/hiện số tuần (NoVerticalHeader = ẩn).
- gridVisible: bật/tắt lưới lịch.
- HorizontalHeaderFormat: định dạng tên ngày
  - SingleLetter / Short / Long / NoHorizontalHeader.

Phương thức:

- selectedDate(): lấy ngày đang chọn (QDate).
- monthShown(), yearShown(): lấy tháng/năm đang hiển thị.
- setFirstDayOfWeek(): đặt ngày bắt đầu tuần.
- selectionChanged(): gọi khi người dùng đổi ngày chọn.

QDate (Qt)

- **QDate** dùng để **xử lý ngày tháng** (năm–tháng–ngày) theo **lịch Gregory**
- Lấy ngày hệ thống và thao tác cộng/trừ, so sánh ngày

Các chức năng chính:

- Lấy ngày: `currentDate()`
- Thiết lập ngày: `setDate(year, month, day)`
- Truy xuất: `year()`, `month()`, `day()`, `dayOfWeek()`
- Cộng ngày/tháng/năm: `addDays()`, `addMonths()`, `addYears()`
- Tính toán: `daysTo()`, `daysInMonth()`, `daysInYear()`
- Kiểm tra năm nhuận: `isLeapYear()`
- Chuyển sang chuỗi: `toPyDate()`

=> Phù hợp để **quản lý, tính toán và hiển thị ngày tháng** trong ứng dụng Qt.

Định dạng ngày tháng trong QDate

- **Ngày**:
  d (1–31), dd (01–31), ddd (Mon…), dddd (Monday…)
- **Tháng**:
  M (1–12), MM (01–12), MMM (Jan…), MMMM (January…)
- **Năm**:
  yy (2 chữ số), yyyy (4 chữ số)

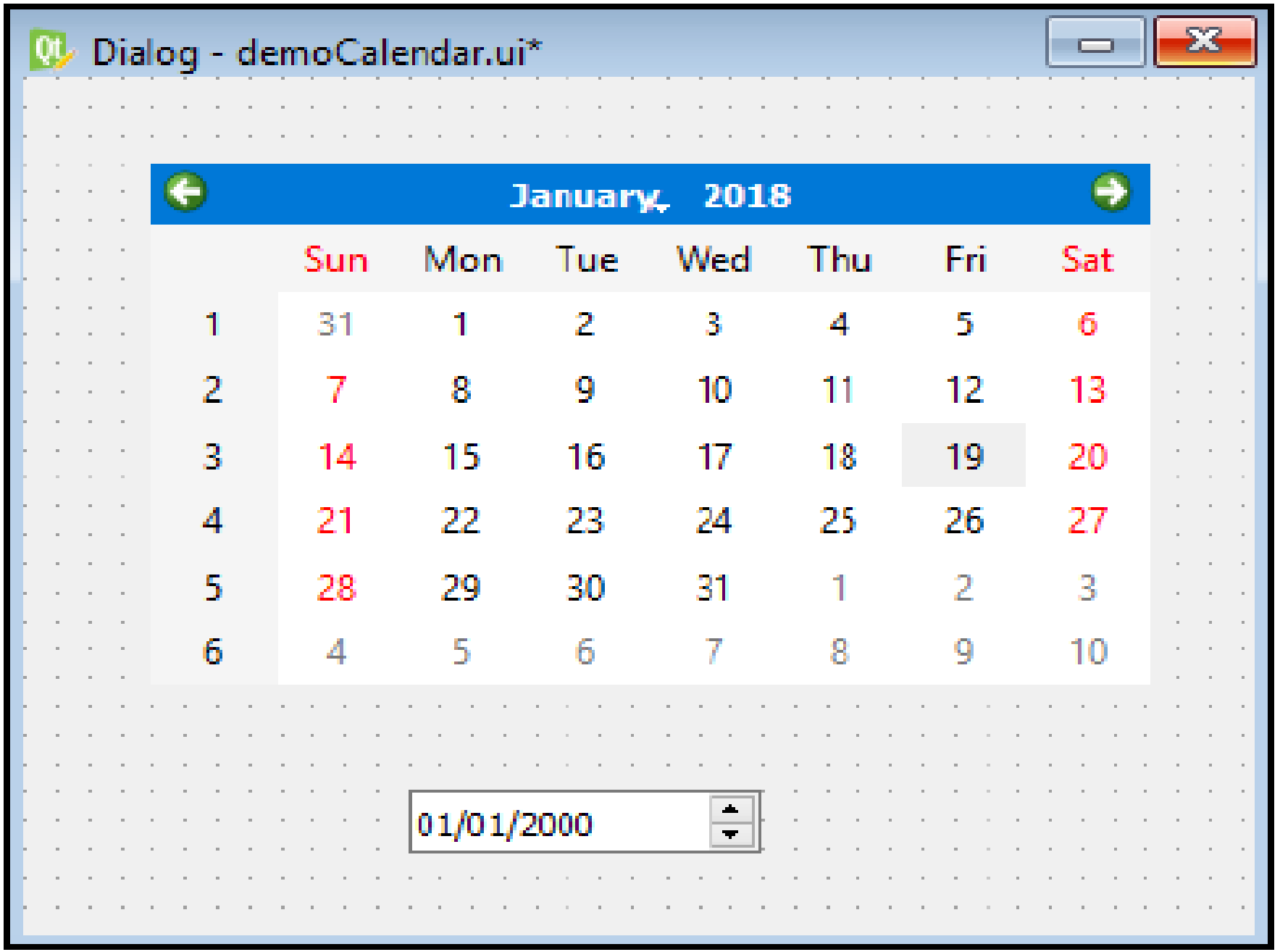
=> Dùng các ký hiệu này để tùy chỉnh cách hiển thị ngày–tháng–năm.

Date Edit Widget (QDateEdit)

- Dùng để **hiển thị và chỉnh sửa ngày tháng**.
- Cho phép **giới hạn ngày chọn**:
  - minimumDate: ngày nhỏ nhất được phép
  - maximumDate: ngày lớn nhất được phép
- Các phương thức chính:
  - setDate(): thiết lập ngày hiển thị
  - setDisplayFormat(): thiết lập **định dạng ngày**
- Một số định dạng ngày:
  - dd.MM.yyyy → 15.01.2018
  - MMM d yy → Jan 15 18
  - MMM d yyyy → Jan 15 2018
  - MMMM d yy → January 15 18

Ứng dụng : Hiển thị dữ liệu khi người dùng chọn từ Calendar Widget

- *Calendar Widget* và *Date Edit Widget* đều dùng để xử lý ngày–tháng–năm.
- Calendar Widget hiển thị dạng lịch lớn, có cả **thứ trong tuần**; Date Edit cho phép **chỉnh ngày/tháng/năm bằng nút xoay**.
- Các bước thực hiện:
  1. Tạo **Dialog without Button** trong Qt Designer.
  2. Thêm **Calendar Widget** và **Date Edit Widget** vào form.
  3. Lưu file demoCalendar.ui và chuyển sang Python bằng **pyuic5**.
  4. Tạo file callCalendar.pyw để gọi giao diện và hiển thị ngày chọn từ Calendar sang Date Edit.


Dialog - demoCalendar.ui*
January 2018
Sun Mon Tue Wed Thu Fri Sat
1 31 1 2 3 4 5 6
2 7 8 9 10 11 12 13
3 14 15 16 17 18 19 20
4 21 22 23 24 25 26 27
5 28 29 30 31 1 2 3
6 4 5 6 7 8 9 10
01/01/2000

Cài đặt ứng dụng: Hiển thị dữ liệu khi người dùng chọn từ Calendar Widget

```
import sys
from PyQt5.QtWidgets import QDialog, QApplication
from demoCalendar import *
class MyForm(QDialog):
    def __init__(self):
        super().__init__()
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)
        self.ui.calendarWidget.selectionChanged.connect
        (self.dispdate)
        self.show()
    def dispdate(self):
        self.ui.dateEdit.setDate(self.ui.calendarWidget.
        selectedDate())
if __name__=="__main__":
    app = QApplication(sys.argv)
    w = MyForm()
    w.show()
    sys.exit(app.exec_())
```

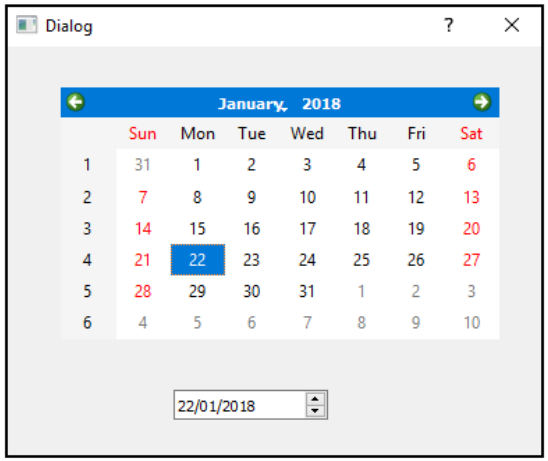


Table Widget (QTableWidget)

Dùng để hiển thị dữ liệu dạng bảng theo **hàng và cột**.

Mỗi ô trong bảng là một **QTableWidgetItem**.

Các phương thức chính:

- `setRowCount()` / `setColumnCount()` : thiết lập số dòng / số cột
- `rowCount()` / `columnCount()` : lấy số dòng / số cột
- `setItem()` : gán dữ liệu cho từng ô
- `clear()` : xóa toàn bộ nội dung bảng

QTableWidgetItem

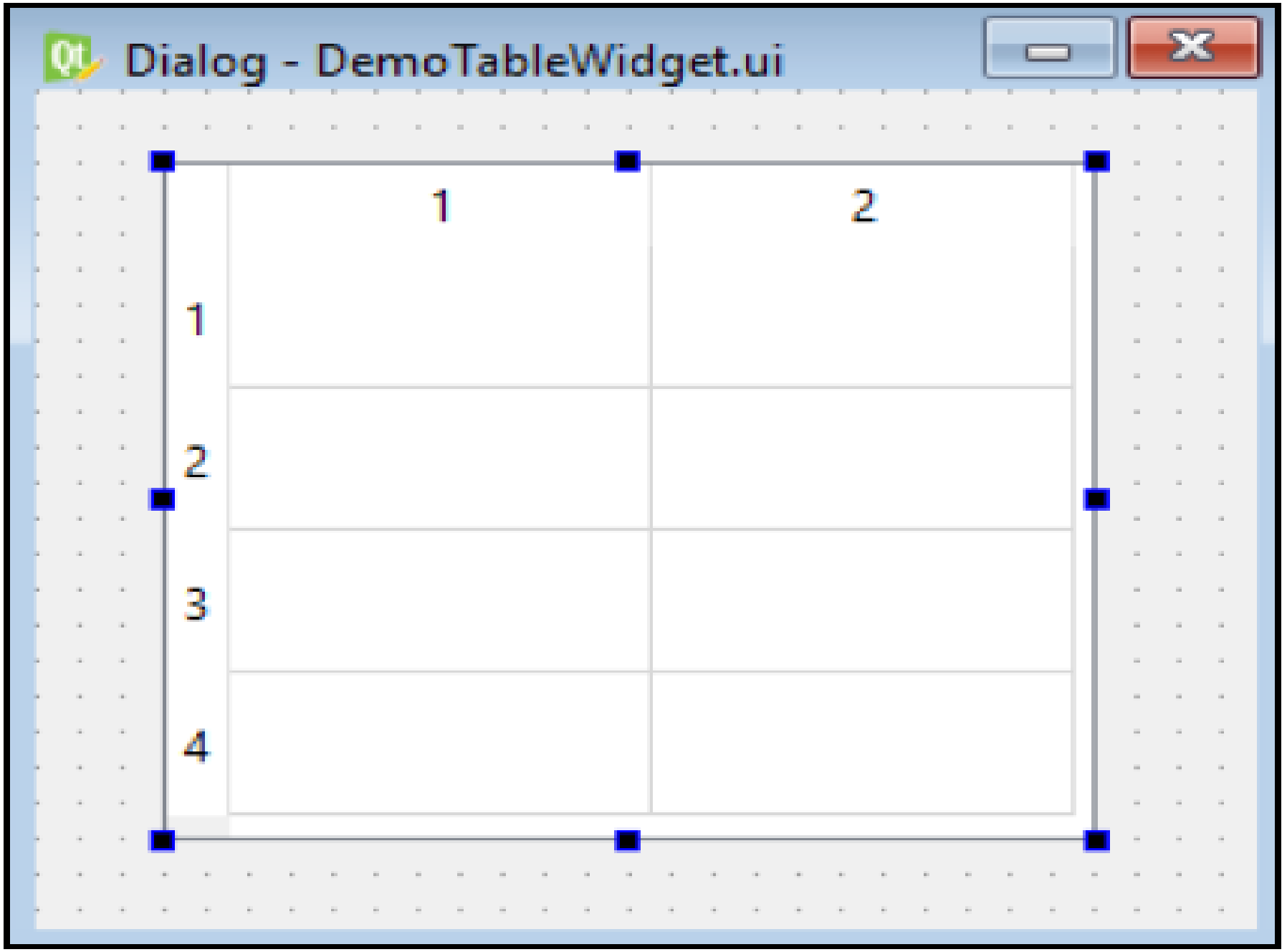
Là lớp đại diện cho từng ô trong **QTableWidget**, dùng để hiển thị nội dung như **văn bản, hình ảnh hoặc widget**.

- `setFont()` : thiết lập kiểu chữ cho nội dung ô
- `setCheckState()` : đánh dấu hoặc bỏ đánh dấu ô
- `checkState()` : kiểm tra ô đang được đánh dấu hay không

Ứng dụng với Table Widget

Các bước hiển thị dữ liệu bằng Table Widget (PyQt):

1. Tạo giao diện mới bằng **Qt Designer** (Dialog without Buttons).
2. Thêm **Table Widget** vào form.
3. Thiết lập số **dòng và cột** (rowCount, columnCount).
4. Bật hiển thị **tiêu đề hàng và cột**.
5. Lưu file `.ui` và chuyển sang **code Python** bằng `pyuic5`.
6. Viết script Python để **gọi giao diện** và **hiển thị dữ liệu** trong Table Widget.


Dialog - DemoTableWidget.ui
1
2
1
2
3
4

Cài đặt ứng dụng Table Widget

```
import sys
from PyQt5.QtWidgets import QDialog, QApplication,QTableWidgetItem
from DemoTableWidget import *
class MyForm(QDialog):
    def __init__(self,data):
        super().__init__()
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)
        self.data=data
        self.addcontent()
    def addcontent(self):
        row=0
        for tup in self.data:
            col=0
        for item in tup:
            oneitem=QTableWidgetItem(item)
            self.ui.tableWidget.setItem(row, col, oneitem)
            col+=1
            row+=1
            data=[]
            data.append(('Suite', '40))
            data.append(('Super Luxury', '30))
            data.append(('Super Deluxe', '20))
            data.append(('Ordinary', '10))
if __name__=="__main__":
    app = QApplication(sys.argv)
    w = MyForm(data)
    w.show()
    sys.exit(app.exec_())
```