QT6 QTable, QMessageBox, Dialogs

QTable

Отображение таблиц. Элементы в QTableWidget предоставляются с помощью QTableWidgetItem.

Методы:

Метод	Назначение
setRowCount()	определения количества строк
setColumnCount()	определения количества столбцов
rowCount()	возвращает количество строк
columnCount()	возвращает количество столбцов

```
from PyQt6.QtWidgets import QApplication, QWidget, QTableWidget, QTableWidgetItem, QVBoxLayout
import sys

class Window(QWidget):
    def __init__(self):
        super().__init__()
        self.setGeometry(200,200, 700, 400)
        self.setWindowTitle("Python GUI Development")
        self.create_button()

    def create_button(self):
        vbox = QVBoxLayout()
        curtable = QTableWidget()
        curtable.setRowCount(3)
        curtable.setColumnCount(3)

        curtable.setItem(0, 0, QTableWidgetItem('Заголовок столбца 1'))
        curtable.setItem(0, 1, QTableWidgetItem('Заголовок столбца 2'))
        curtable.setItem(0, 2, QTableWidgetItem('Заголовок столбца 3'))
```

```
        vbox.addWidget(curtable)

        self.setLayout(vbox)

app = QApplication(sys.argv)
window = Window()
window.show()
sys.exit(app.exec())
```

QMessageBox

QMessageBox - модальный диалог информирования пользователя и получения ответа. В окне сообщения отображается текст, есть необязательный подробный текст в случае необходимости. Также может отображаться значок и стандартные кнопки для принятия ответа пользователя. Существуют различные типы диалогов (about messagebox, information messagebox, warning messagebox, multichoice messagebox).

```
from PyQt6.QtWidgets import QApplication, QDialog, QPushButton
from PyQt6.QtWidgets import QMessageBox

from PyQt6 import uic
import sys

class Window(QDialog):
    def __init__(self):
        super().__init__()
        self.setGeometry(200,200, 700, 400)
        self.setWindowTitle("Python GUI Development")
        uic.loadUi("messagedemo.ui", self)

        self.loadguiobjects()

    def loadguiobjects(self):
        self.butt_warn = self.findChild(QPushButton, "pushButton_warn")
        self.butt_warn.clicked.connect(self.show_warn)
        self.butt_info = self.findChild(QPushButton, "pushButton_info")
        self.butt_info.clicked.connect(self.show_info)
        self.butt_abt = self.findChild(QPushButton, "pushButton_abt")
        self.butt_abt.clicked.connect(self.show_about)

    def show_warn(self):
```

```
        QMessageBox.warning(self, 'Warning', 'This is a warning message')

    def show_info(self):
        #кастомный messagebox + стандартные кнопки
        msg_box = QMessageBox(self)
        msg_box.setWindowTitle('Information')
        msg_box.setText('This is a information message')
        msg_box.setIcon(QMessageBox.Icon.Information)

        # Добавляем кнопки
        ok_button = msg_box.addButton('OK', QMessageBox.ButtonRole.AcceptRole)
        cancel_button = msg_box.addButton('Отменить задание',
QMessageBox.ButtonRole.RejectRole)

        # Показываем сообщение и ждем нажатия кнопки
        msg_box.exec()

        # Проверяем какая кнопка была нажата
        if msg_box.clickedButton() == cancel_button:
            print("Действие отменено")

        elif msg_box.clickedButton() == ok_button:
            print("OK нажата")

    def show_about(self):
        # другой способ с разными кнопкам
        msg_box = QMessageBox(
            QMessageBox.Icon.NoIcon,
            'About',
            'This is a about message',
            QMessageBox.StandardButton.Ok | QMessageBox.StandardButton.Cancel
        )

        # Изменяем текст стандартных кнопок
        msg_box.button(QMessageBox.StandardButton.Ok).setText('Продолжить')
        msg_box.button(QMessageBox.StandardButton.Cancel).setText('Отменить задание')

        result = msg_box.exec()

        if result == QMessageBox.StandardButton.Ok:
```

```
            print("Продолжаем выполнение")
        else:
            print("Задание отменено")

app = QApplication(sys.argv)
window = Window()
window.show()
sys.exit(app.exec())
```

SaveFile

Сама кнопка меню в сформированном из QtDesiner, класс Ui_MainWindow

```
from PyQt6.QtWidgets import QMainWindow, QApplication, QFileDialog, QMessageBox
import sys
from notepadapp import Ui_MainWindow

class NotePadWindow(QMainWindow, Ui_MainWindow):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.show()

        self.actionSave.triggered.connect(self.save_file)

    def save_file(self):
        filename = QFileDialog.getSaveFileName(self, 'Save file')
        if filename[0]:
            f = open(filename[0], 'w')
            with f:
                text = self.textEdit.toPlainText()
                f.write(text)
                QMessageBox.about(self, 'Save file', 'File saved successfully!')

app = QApplication(sys.argv)
Note = NotePadWindow()
sys.exit(app.exec())
```

Revision #3
Created 8 February 2026 16:28:34 by Admin
Updated 10 February 2026 15:51:48 by Admin