Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное   
образовательное учреждение высшего образования   
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №6

Дисциплина: «Операционные системы»

Тема «Применение GUI (graphical user interface) для

клиент-серверного взаимодействия»

Выполнил: Белов А.А.

Группа: 6205-010302D

Самара, 2023

**Задание на лабораторную работу**

Реализовать графический интерфейс пользователя для функционала межпроцессного взаимодействия ранее сделанной лабораторной работы № 4 или для лабораторной работы № 5. В качестве получения дополнительного бонуса допускается реализация обеих лабораторных работ. Все происходящие события должны документироваться: должен писаться лог, то есть журнал событий – сколько процессов запущено, когда, что передано каким клиентом/процессом на сервер и т.п. Должно быть предусмотрено сохранение лога в файл, а также возможность просмотра лога предыдущей запущенной сессии (допустимо использование для этого разных файлов). Ограничений по языку программирования, технологиям, операционной системе нет.

Выполнено GUI на python для 4 лабораторной работы, 5 вариант.

**Текст программы**

Клиентская часть:

import sys  
import socket  
import threading  
import logging  
  
from PyQt5 import QtCore, QtGui, QtWidgets  
  
class Ui\_MainWindow(object):  
 def setupUi(self, Client):  
 Client.setObjectName("Client")  
 Client.resize(550, 400)  
 self.centralwidget = QtWidgets.QWidget(Client)  
 self.centralwidget.setObjectName("centralwidget")  
 self.verticalLayoutWidget = QtWidgets.QWidget(self.centralwidget)  
 self.verticalLayoutWidget.setGeometry(QtCore.QRect(70, 30, 400, 227))  
 self.verticalLayoutWidget.setObjectName("verticalLayoutWidget")  
 self.verticalLayout\_2 = QtWidgets.QVBoxLayout(self.verticalLayoutWidget)  
 self.verticalLayout\_2.setContentsMargins(0, 0, 0, 0)  
 self.verticalLayout\_2.setObjectName("verticalLayout\_2")  
  
 self.label\_4 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(20)  
 self.label\_4.setFont(font)  
 self.label\_4.setStyleSheet("background-color: rgb(255, 0, 4);")  
 self.label\_4.setContextMenuPolicy(QtCore.Qt.NoContextMenu)  
 self.label\_4.setObjectName("label\_4")  
 self.verticalLayout\_2.addWidget(self.label\_4)  
  
 self.pushButton\_3 = QtWidgets.QPushButton(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(15)  
 self.pushButton\_3.setFont(font)  
 self.pushButton\_3.setObjectName("pushButton\_3")  
 self.verticalLayout\_2.addWidget(self.pushButton\_3)  
  
 spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)  
 self.verticalLayout\_2.addItem(spacerItem)  
  
 self.label = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 # font.setFamily("Verdana")  
 font.setPointSize(15)  
 # font.setBold(False)  
 # font.setItalic(False)  
 # font.setUnderline(False)  
 # font.setWeight(50)  
 self.label.setFont(font)  
 self.label.setObjectName("label")  
 self.verticalLayout\_2.addWidget(self.label)  
  
 self.horizontalLayout\_2 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_2.setObjectName("horizontalLayout\_2")  
  
 self.lineEdit = QtWidgets.QLineEdit(self.verticalLayoutWidget)  
 self.lineEdit.setObjectName("lineEdit")  
 self.horizontalLayout\_2.addWidget(self.lineEdit)  
  
 spacerItem1 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)  
 self.horizontalLayout\_2.addItem(spacerItem1)  
  
 self.pushButton = QtWidgets.QPushButton(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(11)  
 self.pushButton.setFont(font)  
 self.pushButton.setObjectName("pushButton")  
 self.horizontalLayout\_2.addWidget(self.pushButton)  
  
 self.verticalLayout\_2.addLayout(self.horizontalLayout\_2)  
  
 self.horizontalLayout\_5 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_5.setObjectName("horizontalLayout\_5")  
  
 self.label\_2 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(13)  
 self.label\_2.setFont(font)  
 self.label\_2.setObjectName("label\_2")  
 self.horizontalLayout\_5.addWidget(self.label\_2)  
  
 self.label\_3 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(13)  
 self.label\_3.setFont(font)  
 self.label\_3.setObjectName("label\_3")  
 self.horizontalLayout\_5.addWidget(self.label\_3)  
  
 self.verticalLayout\_2.addLayout(self.horizontalLayout\_5)  
  
 self.horizontalLayout\_4 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_4.setObjectName("horizontalLayout\_4")  
  
 self.pushButton\_2 = QtWidgets.QPushButton(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(11)  
 self.pushButton\_2.setFont(font)  
 self.pushButton\_2.setObjectName("pushButton\_2")  
 self.horizontalLayout\_4.addWidget(self.pushButton\_2)  
  
 spacerItem2 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)  
 self.horizontalLayout\_4.addItem(spacerItem2)  
  
 self.label\_5 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(13)  
 self.label\_5.setFont(font)  
 self.label\_5.setObjectName("label\_5")  
 self.horizontalLayout\_4.addWidget(self.label\_5)  
  
 self.verticalLayout\_2.addLayout(self.horizontalLayout\_4)  
  
 Client.setCentralWidget(self.centralwidget)  
  
 self.menubar = QtWidgets.QMenuBar(Client)  
 self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 21))  
 self.menubar.setObjectName("menubar")  
 Client.setMenuBar(self.menubar)  
  
 self.statusbar = QtWidgets.QStatusBar(Client)  
 self.statusbar.setObjectName("statusbar")  
 Client.setStatusBar(self.statusbar)  
  
 self.retranslateUi(Client)  
 QtCore.QMetaObject.connectSlotsByName(Client)  
  
 self.pushButton\_3.clicked.connect(self.on\_connect\_server\_clicked)  
 self.pushButton.clicked.connect(self.on\_send\_clicked)  
 self.pushButton\_2.clicked.connect(self.on\_get\_answer\_clicked)  
  
 # Подключение сигнала textChanged к слоту on\_text\_changed  
 self.lineEdit.textChanged.connect(self.on\_text\_changed)  
  
 self.text\_variable = ""  
  
 self.server\_connected = False  
  
 def retranslateUi(self, MainWindow):  
 \_translate = QtCore.QCoreApplication.translate  
 MainWindow.setWindowTitle(\_translate("Client", "Client"))  
 self.label\_4.setText(\_translate("Client", "Сервер не подключён "))  
 self.pushButton\_3.setText(\_translate("Client", "Подключить сервер"))  
 self.label.setText(\_translate("Client", "Введите число со знаком "))  
 self.pushButton.setText(\_translate("Client", "Отправить на сервер"))  
 self.label\_2.setText(\_translate("Client", "Выражение: "))  
 self.label\_3.setText(\_translate("Client", ""))  
 self.pushButton\_2.setText(\_translate("Client", "Получить ответ"))  
 self.label\_5.setText(\_translate("Client", ""))  
  
 #кнопка "подключить/отключить сервер"  
 def on\_connect\_server\_clicked(self):  
 # Логика для обработки нажатия кнопки "Подключить сервер"  
 if self.server\_connected:  
 # Запись лога при отключении сервера  
 logging.info('Client: Сервер отключен')  
  
 # Логика для выключения сервера  
 self.label\_4.setText("Сервер отлючён")  
 self.label\_4.setStyleSheet("background-color: rgb(255, 0, 4);")  
 self.server\_connected = False  
 self.pushButton\_3.setText("Подключить сервер")  
 client\_socket.close()  
  
 else:  
 try:  
 client\_socket.connect((host, port))  
  
 # Запись лога при успешном подключении к серверу  
 logging.info('Client: Клиент подключен к серверу')  
  
 self.label\_4.setText("Сервер подключен")  
 self.label\_4.setStyleSheet("background-color: rgb(0, 255, 21);")  
 self.server\_connected = True  
 self.pushButton\_3.setText("Отключить сервер")  
  
 except OSError as e:  
 # Запись лога при ошибке подключения к серверу  
 logging.error('Client: Ошибка подключения к серверу')  
  
 # Имитация ошибки при подключении к серверу  
 error\_message = "Ошибка подключения к серверу"  
 QtWidgets.QMessageBox.critical(None, "Ошибка", error\_message)  
  
 def on\_text\_changed(self, text):  
 self.text\_variable = text # Слот для обработки изменения текста в QLineEdit  
  
 def on\_send\_clicked(self):  
 # Логика для обработки нажатия кнопки "Отправить на сервер"  
 text = self.lineEdit.text() # Получение текста из QLineEdit  
  
 # Проверка ввода  
 if text != "/0" or text == "/exit" or text == "/clear" or ((text and text[0] in {'+', '-', '/', '\*'} and text[1:].isdigit())):  
 if text == "/exit":  
 self.label\_4.setText("Сервер отключён")  
 self.label\_4.setStyleSheet("background-color: rgb(255, 0, 4);")  
 self.server\_connected = True  
 self.pushButton\_3.setText("Отключить сервер")  
  
 if text == "/0":  
 error\_message = "Ошибка! произошло деление на 0"  
 QtWidgets.QMessageBox.critical(None, "Ошибка", error\_message)  
 self.lineEdit.setFocus() # Установка фокуса на QLineEdit для исправлениz  
 try:  
 self.label\_3.setText(self.text\_variable)  
 client\_socket.send(self.text\_variable.encode()) # Отправка данных серверу  
  
 # Запись лога при отправке данных на сервер  
 logging.debug(f'Client: Отправлены данные на сервер: {self.text\_variable}')  
  
 except OSError as e:  
 self.label\_4.setText("Сервер отлючён")  
 self.label\_4.setStyleSheet("background-color: rgb(255, 0, 4);")  
 error\_message = "Ошибка, сервер не подключён"  
 QtWidgets.QMessageBox.critical(None, "Ошибка", error\_message)  
  
 else:  
 if text == "/0":  
 error\_message = "Ошибка! произошло деление на 0"  
 QtWidgets.QMessageBox.critical(None, "Ошибка", error\_message)  
 self.lineEdit.setFocus() # Установка фокуса на QLineEdit для исправления  
 if not text:  
 self.lineEdit.setFocus() # Установка фокуса на QLineEdit для исправления  
 else:  
 error\_message = "Ошибка, введите сначала знак, а потом число"  
 QtWidgets.QMessageBox.critical(None, "Ошибка", error\_message)  
 self.lineEdit.setFocus() # Установка фокуса на QLineEdit для исправления  
  
 self.lineEdit.clear() # очищаем поле lineEdit  
 self.label\_3.clear() # # очищаем поле label\_3  
 self.label\_5.clear() # очищаем поле label\_5  
  
 def receive\_data(self):  
 response = client\_socket.recv(1024).decode()  
 # Запись лога при получении ответа от сервера  
 logging.debug(f'Client: Получен ответ от сервера: {response}')  
 self.label\_5.setText(response)  
  
 # # Запись лога при получении ответа от сервера  
 # logging.debug(f'Client: Получен ответ от сервера: {response}')  
  
 def on\_get\_answer\_clicked(self):  
 # Логика для обработки нажатия кнопки "Получить ответ"  
 text = "="  
 self.text\_variable = text # Слот для обработки изменения текста в QLineEdit  
 client\_socket.send(self.text\_variable.encode()) # Отправка данных серверу  
 logging.debug(f'Client: Отправка данных серверу: {self.text\_variable}')  
  
 # Создайте и запустите поток для получения данных  
 thread = threading.Thread(target=self.receive\_data)  
 thread.start()  
  
 # result = 0  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 host = '127.0.0.1' # Адрес сервера  
 port = 8009 # Порт сервера  
 client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) # Создание сокета  
  
 logger = logging.getLogger()  
 logger.setLevel(logging.DEBUG)  
  
 # Установка кодировки для логирования  
 log\_encoding = 'utf-8' # Кодировка лог-файла  
 handler = logging.FileHandler('laba\_6.log', encoding=log\_encoding)  
 handler.setFormatter(  
 logging.Formatter('%(asctime)s - %(levelname)s - %(message)s', datefmt='%Y-%m-%d %H:%M:%S'))  
 logger.addHandler(handler)  
  
 logging.info('Client: Клиент запущен')  
  
 app = QtWidgets.QApplication(sys.argv)  
 MainWindow = QtWidgets.QMainWindow()  
 ui = Ui\_MainWindow()  
 ui.setupUi(MainWindow)  
 MainWindow.show()  
 sys.exit(app.exec\_())

Серверная часть:

import sys  
import socket  
import logging  
from threading import Thread  
from PyQt5 import QtCore, QtGui, QtWidgets  
from PyQt5.QtWidgets import QApplication, QMainWindow, QMessageBox  
  
class Ui\_MainWindow(object):  
 def setupUi(self, server):  
 server.setObjectName("server")  
 server.resize(600, 300)  
 self.centralwidget = QtWidgets.QWidget(server)  
 self.centralwidget.setObjectName("centralwidget")  
 self.verticalLayoutWidget = QtWidgets.QWidget(self.centralwidget)  
 # self.verticalLayoutWidget.setGeometry(QtCore.QRect(210, 140, 371, 223))  
 self.verticalLayoutWidget.setGeometry(QtCore.QRect(50,50, 500, 200))  
 self.verticalLayoutWidget.setObjectName("verticalLayoutWidget")  
 self.verticalLayout = QtWidgets.QVBoxLayout(self.verticalLayoutWidget)  
 self.verticalLayout.setContentsMargins(0, 0, 0, 0)  
 self.verticalLayout.setObjectName("verticalLayout")  
  
 self.label = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(20)  
 self.label.setFont(font)  
 self.label.setStyleSheet("background-color: rgb(255, 0, 4);")  
 self.label.setAlignment(QtCore.Qt.AlignCenter)  
 self.label.setObjectName("label")  
 self.verticalLayout.addWidget(self.label)  
  
 self.pushButton = QtWidgets.QPushButton(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(13)  
 self.pushButton.setFont(font)  
 self.pushButton.setObjectName("pushButton")  
 self.verticalLayout.addWidget(self.pushButton)  
  
 spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)  
 self.verticalLayout.addItem(spacerItem)  
  
 self.label\_6 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(17)  
 self.label\_6.setFont(font)  
 self.label\_6.setStyleSheet("background-color: rgb(255, 0, 4);")  
 self.label\_6.setAlignment(QtCore.Qt.AlignCenter)  
 self.label\_6.setObjectName("label\_6")  
 self.verticalLayout.addWidget(self.label\_6)  
  
 spacerItem1 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)  
 self.verticalLayout.addItem(spacerItem1)  
  
 self.horizontalLayout = QtWidgets.QHBoxLayout()  
 self.horizontalLayout.setObjectName("horizontalLayout")  
  
 self.label\_2 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(14)  
 self.label\_2.setFont(font)  
 self.label\_2.setObjectName("label\_2")  
 self.horizontalLayout.addWidget(self.label\_2)  
  
 self.label\_3 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(14)  
 self.label\_3.setFont(font)  
 self.label\_3.setText("")  
 self.label\_3.setObjectName("label\_3")  
 self.horizontalLayout.addWidget(self.label\_3)  
  
 self.verticalLayout.addLayout(self.horizontalLayout)  
  
 self.horizontalLayout\_3 = QtWidgets.QHBoxLayout()  
 self.horizontalLayout\_3.setObjectName("horizontalLayout\_3")  
  
 self.label\_4 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(14)  
 self.label\_4.setFont(font)  
 self.label\_4.setObjectName("label\_4")  
 self.horizontalLayout\_3.addWidget(self.label\_4)  
  
 self.label\_5 = QtWidgets.QLabel(self.verticalLayoutWidget)  
 font = QtGui.QFont()  
 font.setPointSize(14)  
 self.label\_5.setFont(font)  
 self.label\_5.setText("")  
 self.label\_5.setObjectName("label\_5")  
 self.horizontalLayout\_3.addWidget(self.label\_5)  
  
 self.verticalLayout.addLayout(self.horizontalLayout\_3)  
  
 server.setCentralWidget(self.centralwidget)  
 self.menubar = QtWidgets.QMenuBar(server)  
 self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 21))  
 self.menubar.setObjectName("menubar")  
 server.setMenuBar(self.menubar)  
 self.statusbar = QtWidgets.QStatusBar(server)  
 self.statusbar.setObjectName("statusbar")  
 server.setStatusBar(self.statusbar)  
  
 self.retranslateUi(server)  
 QtCore.QMetaObject.connectSlotsByName(server)  
  
 self.pushButton.clicked.connect(self.start\_server)  
  
 self.server\_connected = False  
 self.server\_socket = None  
 self.client\_socket = None  
  
 server.closeEvent = self.closeEvent  
  
 def retranslateUi(self, MainWindow):  
 \_translate = QtCore.QCoreApplication.translate  
 MainWindow.setWindowTitle(\_translate("server", "server"))  
 self.label.setText(\_translate("server", "Сервер выключен"))  
 self.pushButton.setText(\_translate("server", "включить"))  
 self.label\_6.setText(\_translate("server", "Клиент не подключён"))  
 self.label\_2.setText(\_translate("server", "Выражение: "))  
 self.label\_4.setText(\_translate("server", "Результат:"))  
  
 def start\_server(self):  
 self.pushButton.setEnabled(False)  
 self.label.setText("Сервер включен")  
 self.label.setStyleSheet("background-color: rgb(0, 255, 21);")  
 self.server\_connected = True  
  
 # Создание логгера  
 logger = logging.getLogger()  
 logger.setLevel(logging.DEBUG)  
  
 # Установка кодировки для логирования  
 log\_encoding = 'utf-8' # Кодировка лог-файла  
 handler = logging.FileHandler('laba\_6.log', encoding=log\_encoding)  
 handler.setFormatter(  
 logging.Formatter('%(asctime)s - %(levelname)s - %(message)s', datefmt='%Y-%m-%d %H:%M:%S'))  
 logger.addHandler(handler)  
  
 self.server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
 host = '127.0.0.1' # Адрес сервера  
 port = 8009 # Порт сервера  
 self.server\_socket.bind((host, port))  
 self.server\_socket.listen()  
  
 # В отдельном потоке принимаем входящие соединения  
 Thread(target=self.accept\_connections).start()  
  
 def accept\_connections(self):  
 while True:  
 self.client\_socket, client\_address = self.server\_socket.accept()  
  
 self.label\_6.setStyleSheet("background-color: rgb(0, 255, 21);")  
 self.label\_6.setText("Клиент подключился: " + str(client\_address))  
  
 # Запись лога при подключении клиента  
 logging.info(f'Server: Клиент подключился: {client\_address}')  
  
 # В отдельном потоке обрабатываем запросы клиента  
 Thread(target=self.handle\_client).start()  
  
 def handle\_client(self):  
 expression = ""  
 res = 0  
 while True:  
 data = self.client\_socket.recv(1024).decode()  
  
 # Запись лога при получении данных от клиента  
 logging.debug(f'Server: Получены данные от клиента: {data}')  
  
 if not data:  
 continue  
 if data == "/clear":  
 expression = ""  
 res = 0  
 self.label\_3.setText(expression)  
 self.label\_5.setText(str(res))  
 continue  
 if data == "/exit":  
 # Запись лога при отключении клиента  
 logging.info('Server: Клиент отключился')  
  
 self.label\_6.setStyleSheet("background-color: rgb(255, 0, 4);")  
 self.label\_6.setText("Клиент отключился")  
 self.client\_socket.close()  
 self.server\_socket.close()  
  
 expression += data  
 self.label\_3.setText(expression)  
  
 num = data[1:]  
  
 if data[0] == '+':  
 res += int(num)  
 if data[0] == '-':  
 res -= int(num)  
 if data[0] == '\*':  
 res \*= int(num)  
 if data[0] == '/' and data[1] != "0":  
 res /= int(num)  
  
 self.label\_5.setText(str(res))  
 if data == '=':  
 self.client\_socket.send(str(res).encode())  
 expression = ""  
 res = 0  
 self.label\_5.clear()  
 self.label\_3.clear()  
  
 def closeEvent(self, event):  
 if self.server\_connected:  
 message\_box = QMessageBox(self.centralwidget)  
 message\_box.setWindowTitle("Закрытие сервера")  
 message\_box.setText("Сервер будет закрыт. Вы уверены?")  
  
 message\_box.setStandardButtons(QMessageBox.Yes | QMessageBox.No)  
 message\_box.setDefaultButton(QMessageBox.No)  
  
 reply = message\_box.exec\_()  
 if reply == QMessageBox.Yes:  
 # Запись лога при закрытии сервера  
 logging.info('Server: Сервер закрыт')  
 if self.client\_socket:  
 self.client\_socket.close()  
 if self.server\_socket:  
 self.server\_socket.close()  
 event.accept()  
 else:  
 event.ignore()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 app = QtWidgets.QApplication(sys.argv)  
 MainWindow = QtWidgets.QMainWindow()  
 ui = Ui\_MainWindow()  
 ui.setupUi(MainWindow)  
 MainWindow.show()  
 sys.exit(app.exec\_())