**Лабораторная работа №6**

**Организация взаимодействия с базой данных через приложение с графическим интерфейсом**

**Цель работы:** получение навыков разработки приложений для взаимодействия с базой данных, содержащих графический интерфейс пользователя.

**Основные теоретические сведения**

Графический пользовательский интерфейс – это система средств для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана.

Программы с графическим интерфейсом пользователя событийно-ориентированные. В событийно-ориентированном программировании та или иная часть программного кода начинает выполняться лишь тогда, когда случается то или иное событие. Оно базируется на объектно-ориентированном и структурном программировании. Даже если мы не создаем собственных классов и объектов, то все-равно ими пользуемся. Все виджеты – это объекты, порожденные встроенными классами.

Так, например, в языке python для реализации «оконных» приложений используется библиотека Tkinter (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Tk interface). Это кросс-платформенная событийно-ориентированная графическая библиотека на основе средств Tk. Входит в состав стандартной библиотеки языка Python. Tkinter – это свободное программное обеспечение, распространяемое под python-лицензией

Библиотека предназначена для организации диалогов в программе с помощью оконного графического интерфейса ([GUI](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F)). Под графическим интерфейсом пользователя (GUI) подразумевается совокупность виджетов – окон, кнопок, текстовых полей для ввода, скроллеров, списков, радиокнопок, флажков и др., которые пользователь видит на экране, открывая то или иное приложение. Через них пользователь взаимодействует с программой и управляет ею.

Для написания программы с графическим интерфейсом необходимо выполнение следующих действий:

1. Создать главное окно;

2. Создать виджеты и выполнить конфигурацию их свойств (опций);

3. Определить события (то, на что будет реагировать программа);

4. Описать обработчики событий (то, как будет реагировать программа);

5. Расположить виджеты в главном окне;

6. Запустить цикл обработки событий.

**Задание к работе**

1. Изучить библиотеку для реализации приложения с графическим интерфейсом на выбранном языке программирования.

2. Разработать приложение с графическим интерфейсом, которое обеспечит подключение к базе данных, разработанной на основе предыдущих лабораторных работ, а также обеспечит выполнение запросов.

**Пример выполнения задания**

**import** sqlite3  
**import** tkinter **as** tk *# Модули пакета импортируются отдельно*  
**from** tkinter **import** ttk  
**from** tkinter **import** messagebox **as** mb  
**from** tkinter **import** filedialog **as** fd  
**from** typing **import** List, Optional, Tuple, Callable  
**from** functools **import** partial

Описание главного экрана графического приложения представлено классом Main. В нем располагаются кнопки меню, а также выводятся запрашиваемые из базы данных данные.

**class** Main(tk.Frame):  
 *"""Главный экран графического приложения"""* **def** \_\_init\_\_(self, root):  
 super().\_\_init\_\_(root)  
 self.\_\_db = ServiceDB(**'sql.db'**) *# Объект БД*

*# Табличное представление* self.tree: Optional[ttk.Treeview] = None  *# Пункты меню и функции получения данных идентичны примеру из лабораторной работы №5* self.\_\_entities = {  
 1: **'Заказчики'**,  
 2: **'Исполнители'**,  
 3: **'Квалификации'**,  
 4: **'Договоры подряда'**,  
 5: **'Виды работ'**,  
 6: **'Статусы договора'** }

self.menu = {  
 1: (  
 **lambda**: self.\_\_db.execute\_select(  
 **'customer'**,  
 fields=[**'contact.fcs'**, **'contact.phone'**, **'contact.address'**],  
 joins=[[**'contact'**, **'contact'**, **'contact.id = contact\_id'**]],  
 order\_by=**'-contact.fcs'** ),  
 [**'ФИО'**, **'Телефон'**, **'Адрес'**]  
 ),

Tkinter предоставляет большое количество графических элементов, которые мы можем настраивать и размешать на экране приложения. Код ниже создает на экране кнопки меню. При нажатии на кнопку вызывается метод \_\_show\_result(), выводящий табличное представление данных, хранящихся в соответствующей таблице базы данных.

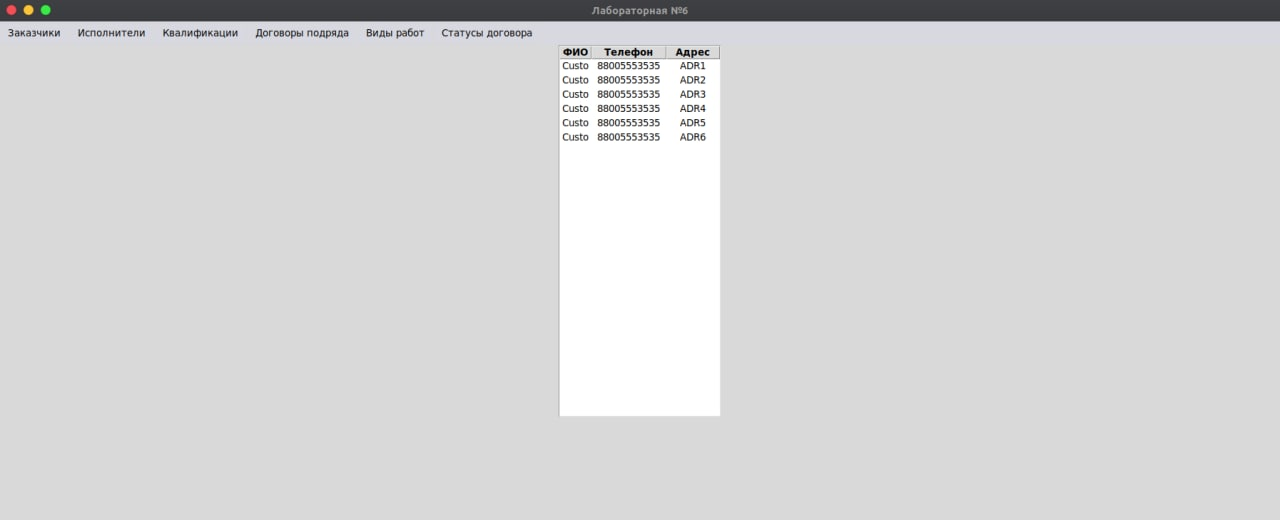
*# Панель с кнопками меню*toolbar = tk.Frame(bg=**'#d7d8e0'**, bd=2)  
**for** option, name **in** self.\_\_entities.items():  
 *# Создаём каждый объект и упаковываем их на toolbar* tk.Button(  
 toolbar,  
 text=name,  
 *# Функция, которая выполняется при нажатии на кнопку.  
 # Это функция вывода таблицы, и кортеж из функции получения данных и имён столбцов таблицы* command=partial(self.\_\_show\_result, self.menu[option]),  
 bg=**'#d7d8e0'**,  
 bd=0,  
 compound=tk.TOP,  
 padx=10  
 ).pack(side=tk.LEFT)  
*# Устанавливаем toolbar на экран*toolbar.pack(side=tk.TOP, fill=tk.X)  
*# Определяем рабочее пространство*self.main\_workplace = tk.Frame()  
self.main\_workplace.pack(side=tk.TOP, fill=tk.X)

Метод \_\_show\_result() принимает на вход функцию, получающую из базы данных список записей и заголовки столбцов для таблицы. Если таблица уже отображена на экране, метод её очищает, а затем создает новую.

**def** \_\_show\_result(self, params: Tuple[Callable, List[str]]):  
 func, columns = params  
  
 *# Если таблица уже была - уничтожаем её* **if** self.tree **is not** None:  
 self.tree.destroy()  
 *# Создаём новую таблицу* self.tree = ttk.Treeview(self.main\_workplace, columns=columns, height=25, show=**'headings'**)  
  
 *# Заполняем каждый столбец именами столбцов* **for** column **in** columns:  
 self.tree.column(column, width=len(column) \* 15, anchor=tk.CENTER)  
 self.tree.heading(column, text=column)  
 self.tree.pack(side=tk.TOP)  
  
 *# Заполняем таблицу данными* **for** row **in** func():  
 self.tree.insert(**''**, **'end'**, values=row)

При запуске программы создается экземпляр приложения, задаются базовые настройки, после чего приложение выводится на экран.

**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 root = tk.Tk()  
 app = Main(root)  
 app.pack()  
 root.title(**"Лабораторная №6"**)  
 root.geometry(**"1800x700"**)  
 root.mainloop()



Изображение выглядит как стол

Автоматически созданное описание

Изображение выглядит как стол

Автоматически созданное описание

Рис. 5. Снимки экрана выполнения программы

В процессе выполнения лабораторной работы были получены навыки разработки приложения для взаимодействия с базой данных, с использованием графического интерфейса.