РАЗРАБОТКА ПРИЛОЖЕНИЯ В СРЕДЕ ПРОГРАММИРОВАНИЯ C# ДЛЯ РАБОТЫ С БАЗАМИ ДАННЫХ ПРЕДПРИЯТИЯ

Николюкин М.С.
Технический колледж ГОУ ВПО «Тамбовский государственный технический университет»
Тамбов, Россия

В современном мире информационные технологии стали неотъемлемой частью нашей действительности. Их развитие стало одним из приоритетных направлений во всех сферах жизнедеятельности. Не осталась в стороне и предпринимательская деятельность. Всё больше и больше компаний нуждаются в специализированных продуктах учёта клиентской базы, особенно это актуально для фирм занимающихся Интернет-торговлей.

В данной статье я хочу показать пример разработки такого продукта на примере одного канцелярского магазина.

Для этой цели я выбрал среду программирования C#, т.к. C# - оптимальный инструмент для создания приложений любой сложности. Оптимальный, т.к. поддерживает технологию визуальной разработки, которая позволяет существенно сократить время разработки (снизить стоимость, соответственно), при сохранении хорошего качества и надежности программного продукта.

Плюсы языка:

- Кроссплатформенность;
- Быстрая скорость разработки;
- Быстродействие;
- Тысячи готовых к употреблению классов, реализующие различные алгоритмы, сокращают сроки разработки и повышают надежность программ;

Передо мной была поставлена следующая задача:

Разработать приложение работы с базами данных юридических лиц для магазина канцелярских товаров. Приложение должно поддерживать следующие режимы:

- Внесение данных об организации ;
- Сохранение и поиск внесенной информации;

В соответствии с этим на главном окне приложения должны находиться поля для ввода наименования организации, юридического адреса, города, телефона, количества товара. Также должен присутствовать комбинированный список, в котором находятся наименования товаров, имеющиеся в магазине на данный момент.

Окно приложения должно обеспечивать необходимые функции системы, связи между ее компонентами, его модель представлена на рисунке 1:

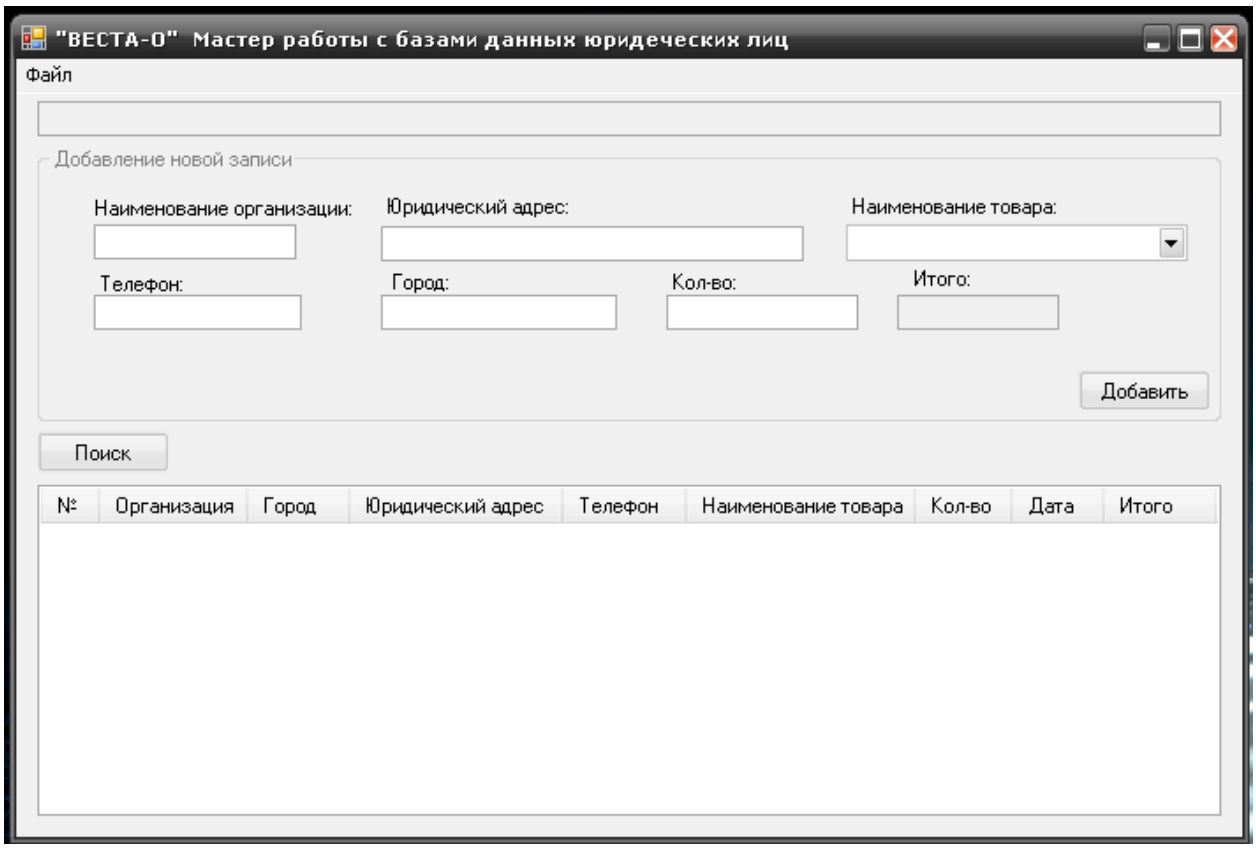

Рисунок 1 – Главное окно программы

Главное окно программы разделено на две области. Первая область предусматривает занесение основных сведений об организации и подсчёта итоговой стоимости, исходя из выбранного товара и его количества. Во второй располагается кнопка поиска и таблица для вывода содержимого файла базы данных.

Для начала работы с базой, пользователю необходимо либо открыть уже существующую базу, либо создать новую, используя меню «Файл».Так для добавления записи об организации в базу необходимо ввести соответствующие данные в поля ввода. При нажатии кнопки «Добавить» происходит запись введенных данных в структуру, а затем и в файл.

В программе предусмотрен поиск информации по названию организации и дате. Для этого необходимо нажать кнопку «Поиск», открывшемся окне, необходимо ввести критерии поиска и нажать кнопку «Поиск». Если необходимо совершить поиск по другим критериям, то в программе предусмотрена кнопка «Очистка», которая позволяет очистить результаты предыдущего поиска. Окно поиска представлено на рисунке 2.

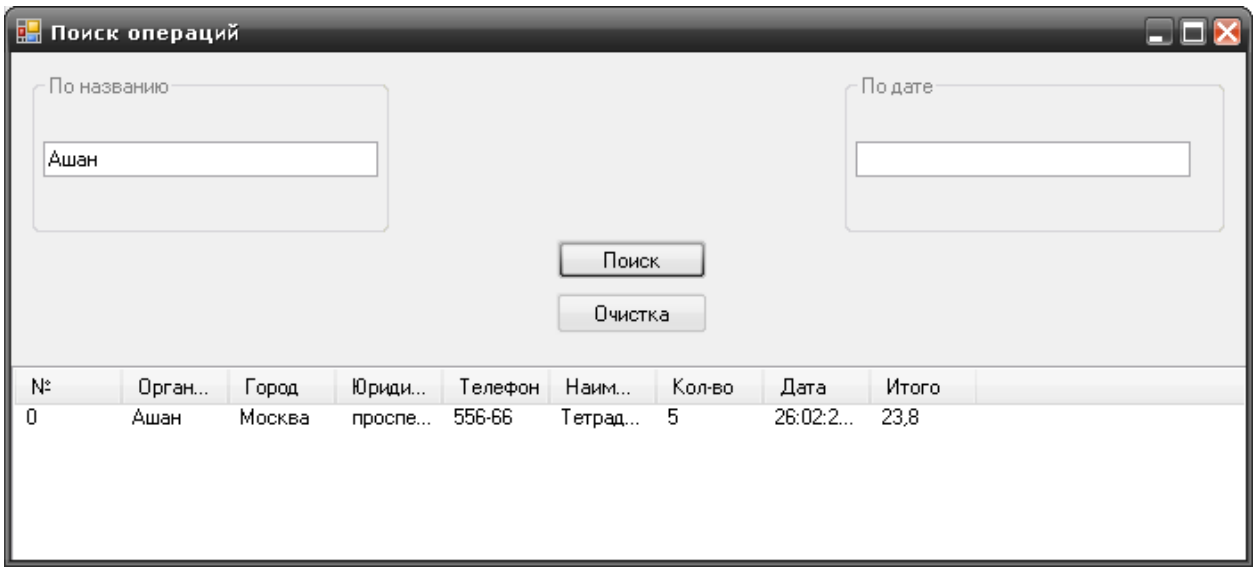

Рисунок 2 – Главное окно программы

Для реализации описанной модели необходимо выделить необходимые переменные, обеспечить механизмы их взаимодействия и функции обработки данных.

В данной программе используются следующие переменные:

filestruct – struct

index – int

price – double

В структуру filestruct входят следующие переменные

num – string

name – string

city– string

address– string

phone– string

item– string

number– string

date – string

al l– string

В переменную num записывается порядковый номер записи, в name – название организации, в city – город, в address – юридический адрес, в phone – телефон, в item – выбранный товар из комбинированного списка, в number – количество товара, в date – дата операции, в all – итоговая стоимость . По нажатию кнопки

«Добавить» происходит внесение введенных данных в структуру, файл, а из файла эти данные загружаются в таблицу.

```
File.AppendAllText(txtFile.Text, num.ToString() + Environment.NewLine);
File.AppendAllText(txtFile.Text, txtName.Text + Environment.NewLine);
File.AppendAllText(txtFile.Text, txtCity.Text + Environment.NewLine);
File.AppendAllText(txtFile.Text, txtAdres.Text + Environment.NewLine);
File.AppendAllText(txtFile.Text, txtPhone.Text + Environment.NewLine);
File.AppendAllText(txtFile.Text, cmbItems.SelectedItem.ToString() + Environment.NewLine);
File.AppendAllText(txtFile.Text, txtNumber.Text + Environment.NewLine);
File.AppendAllText(txtFile.Text, DateTime.Now.ToString("dd:MM:yyyy") + Environment.NewLine);

listView1.Items.Add(num.ToString());
listView1.Items[index].SubItems.Add(txtName.Text);
listView1.Items[index].SubItems.Add(txtCity.Text);
listView1.Items[index].SubItems.Add(txtAdres.Text);
listView1.Items[index].SubItems.Add(txtPhone.Text);
listView1.Items[index].SubItems.Add(cmbItems.SelectedItem.ToString());
listView1.Items[index].SubItems.Add(txtNumber.Text);
listView1.Items[index].SubItems.Add(DateTime.Now.ToString("dd:MM:yyyy"));
```

При создании новой базы создаётся пустой файл в который буду записываться элементы структуры.

```
OpenFileDialog openFile = new OpenFileDialog();
openFile.Filter = "Файл БД (*.ves)|*.ves";
if (openFile.ShowDialog() == DialogResult.OK)
{
   txtFile.Text = openFile.FileName;
   fn = txtFile.Text;
   id = 1;
}
```

При открытии уже существующей базы, приложение сначала обращается к вспомогательному файлу с расширением *.vesindex и считывает конечное количество записей в таблице, а затем обращается к файлу с самой базой данных и с помощью цикла начинает считывать записи в структуру и поочередно добавлять их в таблицу.

```
StreamReader findex = new StreamReader(txtFile.Text+"index");
num = int.Parse(findex.ReadLine());
index = int.Parse(findex.ReadLine());

StreamReader basef = new StreamReader(txtFile.Text);
for (i = 0; i <=num; i++)
{
   filestr.num = basef.ReadLine();
   filestr.name = basef.ReadLine();
   filestr.city = basef.ReadLine();
   filestr.address = basef.ReadLine();
   filestr.phone = basef.ReadLine();
   filestr.item = basef.ReadLine();
   filestr.number = basef.ReadLine();
   filestr.date = basef.ReadLine();
   filestr.all = basef.ReadLine();

   listView1.Items.Add(i.ToString());
   listView1.Items[i].SubItems.Add(filestr.name);
   listView1.Items[i].SubItems.Add(filestr.city);
   listView1.Items[i].SubItems.Add(filestr.address);
   listView1.Items[i].SubItems.Add(filestr.phone);
   listView1.Items[i].SubItems.Add(filestr.item);
   listView1.Items[i].SubItems.Add(filestr.number);
   listView1.Items[i].SubItems.Add(filestr.date);
   listView1.Items[i].SubItems.Add(filestr.all);

}
```

По нажатию кнопки «Поиск» появляется форма поиска, в которой пользователь может выбрать 2 критерия поиска:

- По дате;
- По названию организации;

При вводе поискового запроса программа обращается ко всем записям в файле, записывая их в структуру, сверяя – встречается ли запись соответствующая запись в базе. Если встречается – результат выводится в таблицу, если нет – сообщение «Ничего не найдено». Для перехода к следующему поиску – пользователю необходимо очистить таблицу, с помощью кнопки «Очистка».