САНКТ-ПЕТЕРБУРГСКОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАНИЕ   
«КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

**ОТЧЕТ**

**по практической работе №2**

**«Связные списки»**

Выполнил студент 41 гр.

Березко В.В.

Преподаватель: Фомин А. В.

Оценка: \_\_\_\_\_\_\_\_\_

Санкт-Петербург 2016

**Оглавление**

[1. Цель работы 3](#_Toc467707416)

[2. Ход выполнения работы 4](#_Toc467707417)

[3. Заключение 7](#_Toc467707418)

# Цель работы

Закрепление на конкретных примерах полученных теоретических знаний при изучении свойств, методов и событий стандартных компонентов: простого списка ListBox и комбинированного списка ComboBox. Изучить способ поиска нужного файла в текущей папке. Создание на основе изученной теории приложения по выбору необходимой информации.

# Ход выполнения работы

Создаем форму и размещаем на ней comboBox для отображения групп, listBox для отображения свойств групп, два textBox для добавления новых групп и свойств, 5 кнопок для управления свойствами и сохранения результата работы и checkBox для включения и отключения сортировки listBox.

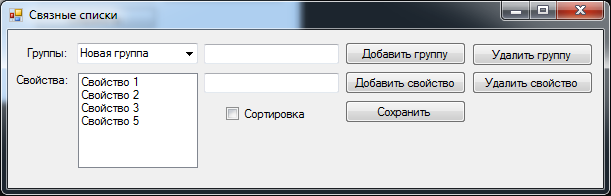


Рисунок - Главная форма приложения

Для сохранения списков в памяти программы необходимо создать список списков строк:

List<string> list = new List<string>();

List<List<string>> link = new List<List<string>>();

Для загрузки групп и свойств из файлов были создано несколько функций:

private void Form1\_Load(object sender, EventArgs e)

{

loadList("../../list.txt");

for (int i = 0; i < list.Count; i++)

comboBox1.Items.Add(list[i]);

}

void loadList(string file)

{

StreamReader read = new StreamReader(file);

string count = read.ReadLine();

int c = int.Parse(count);

for (int i = 0; i < c; i++)

{

list.Add(read.ReadLine());

loadLinks(i, "../../" + list[i] + ".txt");

}

read.Close();

}

void loadLinks(int n, string file)

{

try

{

StreamReader read = new StreamReader(file);

string count = read.ReadLine();

int c = int.Parse(count);

List<string> l = new List<string>();

for (int i = 0; i < c; i++)

l.Add(read.ReadLine());

link.Add(l);

read.Close();

}

catch (Exception e)

{

list.RemoveAt(n);

}

}

Для добавления и удаления свойств на события Click кнопок были написаны следующие функции:

// Удалить свойство

private void button2\_Click(object sender, EventArgs e)

{

if(listBox1.SelectedIndex != -1)

{

link[comboBox1.SelectedIndex].RemoveAt(listBox1.SelectedIndex);

listBox1.Items.RemoveAt(listBox1.SelectedIndex);

nsave = true;

}

}

// Добавить свойство

private void button4\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "")

{

listBox1.Items.Add(textBox1.Text);

link[comboBox1.SelectedIndex].Add(textBox1.Text);

textBox1.Text = "";

nsave = true;

}

}

// Удалить группу

private void button3\_Click(object sender, EventArgs e)

{

if (comboBox1.SelectedIndex != -1)

{

link.RemoveAt(comboBox1.SelectedIndex);

comboBox1.Items.RemoveAt(comboBox1.SelectedIndex);

listBox1.Items.Clear();

nsave = true;

}

}

// Добавить группу

private void button5\_Click(object sender, EventArgs e)

{

if(textBox2.Text != "")

{

comboBox1.SelectedIndex = comboBox1.Items.Add(textBox2.Text);

link.Add(new List<string>());

list.Add(textBox2.Text);

textBox2.Text = "";

nsave = true;

}

}

# Заключение

Благодаря проделанной работе, я закрепил на конкретных примерах теоретические знания, полученные при изучении свойств, методов и событий стандартных компонентов: простого и комбинированного списка. Было разработано приложение для создания списков со свойствами с возможностью создания, редактирования, изменения и сохранения списков.