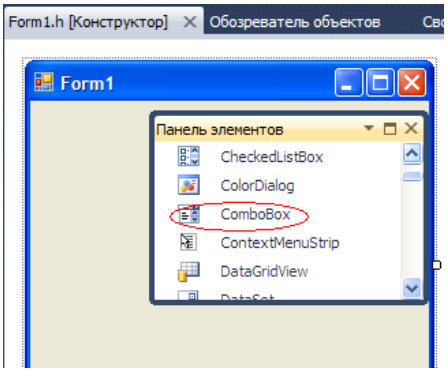
Лабораторная работа 2. Управляющие элементы. Работа с компонентами списками ListBox, ComboBox

Цель работы: разработать приложение с применением управляющих элементов работы со списками: ListBox, ComboBox

Элемент ComboBox

Элемент ComboBox образует выпадающий список и совмещает функциональность компонентов ListBox и TextBox.



Для хранения элементов списка в ComboBox также предназначено свойство **Items**.

Подобным образом, как и с ListBox, можно в окне свойств на свойство Items и нам отобразится окно для добавления элементов ComboBox:

И как и с компонентом ListBox, здесь мы также можем программно управлять элементами.

Добавление элементов

При добавлении с помощью методов Add / AddRange все новые элементы помещаются в конец списка. Однако если мы зададим у ComboBox свойство Sorted равным true, тогда при добавлении будет автоматически производиться сортировка.

Удаление элементов

Можно получить элемент по индексу и производить с ним разные действия. Например, изменить его: comboBox1.Items[0] = "Парагвай";

- **Настройка оформления ComboBox**
- С помощью ряда свойств можно настроить стиль оформления компонента. Так, свойство **DropDownWidth** задает ширину выпадающего списка. С помощью свойства **DropDownHeight** можно установить высоту выпадающего списка.
- Еще одно свойство

MaxDropDownItems позволяет задать число видимых элементов списка - от 1 до 100. По умолчанию это число равно 8.

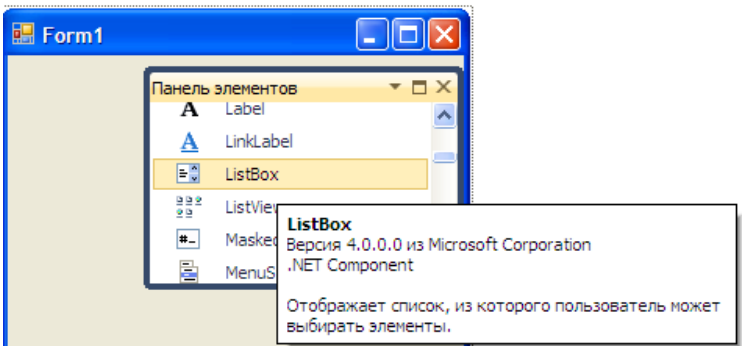
Свойство **DropDownStyle** задает стиль ComboBox. Оно может принимать три возможных значения:

- **Dropdown**: используется по умолчанию. Мы можем открыть выпадающий список вариантов при вводе значения в текстовое поле или нажав на кнопку со стрелкой в правой части элемента, и нам отобразится собственно выпадающий список, в котором можно выбрать возможный вариант
- **DropdownList**: чтобы открыть выпадающий список, надо нажать на кнопку со стрелкой в правой стороне элемента
- **Simple**: ComboBox представляет простое текстовое поле, в котором для перехода между элементами мы можем использовать клавиши клавиатуры вверх/вниз
- Элемент ComboBox образует выпадающий список и совмещает функциональность компонентов ListBox и TextBox. Для хранения элементов списка в ComboBox также предназначено свойство **Items**.
- Подобным образом, как и с ListBox, можно в окне свойств на свойство Items и нам отобразится окно для добавления элементов ComboBox:
- И как и с компонентом ListBox, здесь мы также можем программно управлять элементами.
- Добавление элементов
- При добавлении с помощью методов Add / AddRange все новые элементы помещаются в конец списка. Однако если мы зададим у ComboBox свойство Sorted равным true, тогда при добавлении будет автоматически производиться сортировка.
- Удаление элементов

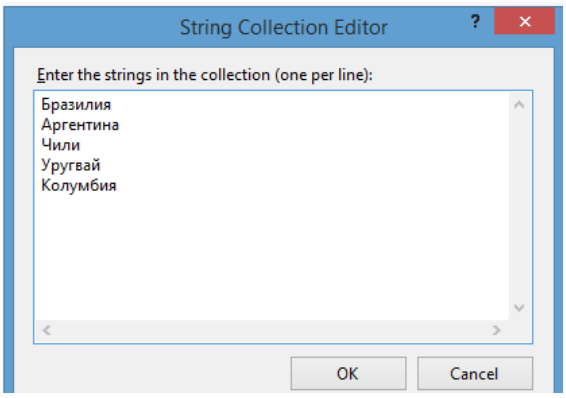
- Можно получить элемент по индексу и производить с ним разные действия. Например, изменить его:
- comboBox1.Items[0] = "Парагвай";
- **Настройка оформления ComboBox**
- С помощью ряда свойств можно настроить стиль оформления компонента. Так, свойство **DropDownWidth** задает ширину выпадающего списка. С помощью свойства **DropDownHeight** можно установить высоту выпадающего списка.
- Еще одно свойство
- **MaxDropDownItems** позволяет задать число видимых элементов списка - от 1 до 100. По умолчанию это число равно 8.
- Свойство **DropDownStyle** задает стиль ComboBox. Оно может принимать три возможных значения:
- **Dropdown**: используется по умолчанию. Мы можем открыть выпадающий список вариантов при вводе значения в текстовое поле или нажав на кнопку со стрелкой в правой части элемента, и нам отобразится собственно выпадающий список, в котором можно выбрать возможный вариант
- **DropdownList**: чтобы открыть выпадающий список, надо нажать на кнопку со стрелкой в правой стороне элемента
- **Simple**: ComboBox представляет простое текстовое поле, в котором для перехода между элементами мы можем использовать клавиши клавиатуры вверх/вниз

ListBox

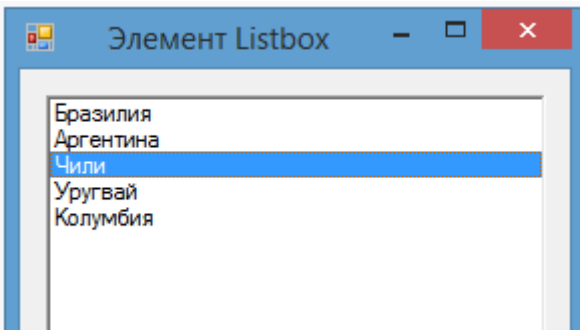
Элемент ListBox представляет собой простой список. Ключевым свойством этого элемента является свойство Items, которое как раз и хранит набор всех элементов списка.



Элементы в список могут добавляться как во время разработки, так и программным способом. В Visual Studio в окне Properties (Свойства) для элемента ListBox мы можем найти свойство Items. После двойного щелчка на свойство нам отобразится окно для добавления элементов в список



В пустое поле мы вводим по одному элементу списка - по одному на каждой строке. После этого все добавленные нами элементы окажутся в списке, и мы сможем ими управлять:



Программное управление элементами в ListBox

Добавление элементов

Итак, все элементы списка входят в свойство Items, которое представляет собой коллекцию. Для добавления нового элемента в эту коллекцию, а значит и в список, надо использовать метод Add, например: listBox1.Items.Add("Новый элемент");. При использовании этого метода каждый добавляемый элемент добавляется в конец списка.

Можно добавить сразу несколько элементов, например, массив. Для этого используется метод AddRange:

```
string[] countries = { "Бразилия", "Аргентина", "Чили", "Уругвай", "Колумбия" };
listBox1.Items.AddRange(countries);
```

Вставка элементов

В отличие от простого добавления вставка производится по определенному индексу списка с помощью метода Insert:

```
1listBox1.Items.Insert(1, "Парагвай");
```

В данном случае вставляем элемент на вторую позицию в списке, так как отсчет позиций начинается с нуля.

Удаление элементов

Для удаления элемента по его тексту используется метод Remove:

```
1listBox1.Items.Remove("Чили");
```

Чтобы удалить элемент по его индексу в списке, используется метод RemoveAt:

```
1listBox1.Items.RemoveAt(1);
```

Кроме того, можно очистить сразу весь список, применив метод Clear:

```
1listBox1.Items.Clear();
```

Доступ к элементам списка

Используя индекс элемента, можно сам элемент в списке. Например, получим первый элемент списка:

```
1string firstElement = listBox1.Items[0];
```

Метод Count позволяет определить количество элементов в списке:

```
1int number = listBox1.Items.Count();
```

Выделение элементов списка

При выделении элементов списка мы можем ими управлять как через индекс, так и через сам выделенный элемент. Получить выделенные элементы можно с помощью следующих свойств элемента ListBox:

- SelectedIndex: возвращает или устанавливает номер выделенного элемента списка. Если выделенные элементы отсутствуют, тогда свойство имеет значение -1
- SelectedIndices: возвращает или устанавливает коллекцию выделенных элементов в виде набора их индексов
- **SelectedItem**: возвращает или устанавливает текст выделенного элемента
- SelectedItems: возвращает или устанавливает выделенные элементы в виде коллекции

По умолчанию список поддерживает выделение одного элемента. Чтобы добавить возможность выделения нескольких элементов, надо установить у его свойства SelectionMode значение MultiSimple.

Чтобы выделить элемент програмно, надо применить метод SetSelected(int index, bool value), где index - номер выделенного элемента. Если второй параметр - value имеет значение true, то элемент по указанному индексу выделяется, если false, то выделение наоборот скрывается:

```
1listBox1.SetSelected(2, true); // будет выделен третий элемент
```

Чтобы снять выделение со всех выделенных элементов, используется метод ClearSelected.

Контрольные вопросы:

1. Назовите основные свойства управляющего элемента ComboBox
2. Назовите основные управляющего элемента **ListBox**