Delphi. Тема 11:

«Проектирование окон диалога с управляющими элементами»

План темы:

1. Назначение диалоговых окон и управляющих элементов.
2. Группировка управляющих элементов.
3. Редактор с шаблоном.
4. Кнопки для увеличения и уменьшения числовых значений.
5. Список.
6. Выпадающий список.

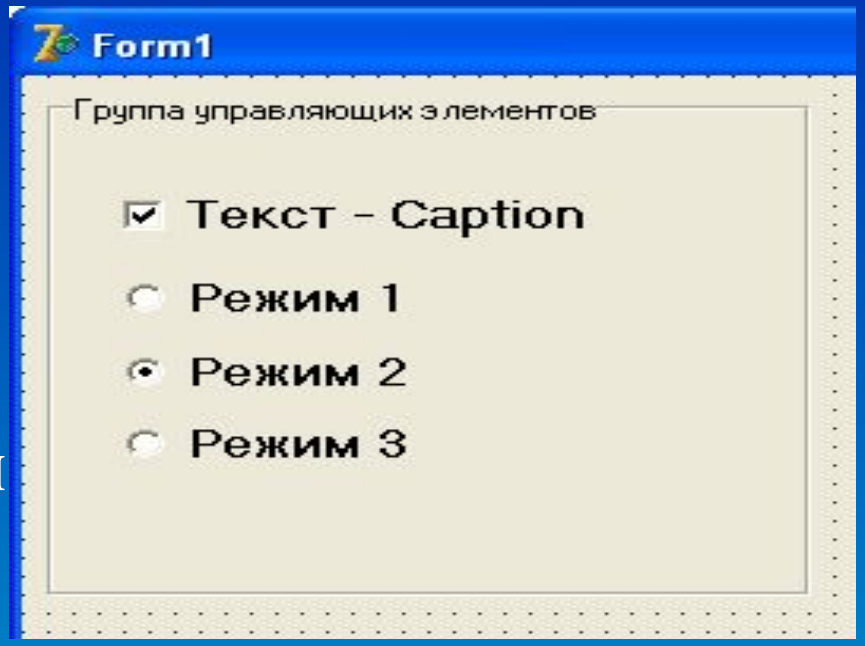
1. Назначение диалоговых окон и управляющих элементов.

- **В основе диалога программы с пользователем лежит *окно диалога* - вспомогательное окно фиксированного размера, содержащее различные управляющие элементы: кнопки, строки редактирования, независимые и зависимые переключатели, списки и т. д. С помощью управляющих элементов пользователь просматривает и вводит данные, а также управляет диалогом. В среде Delphi окно диалога создается на основе обычной формы.**

2. Группировка управляющих элементов.

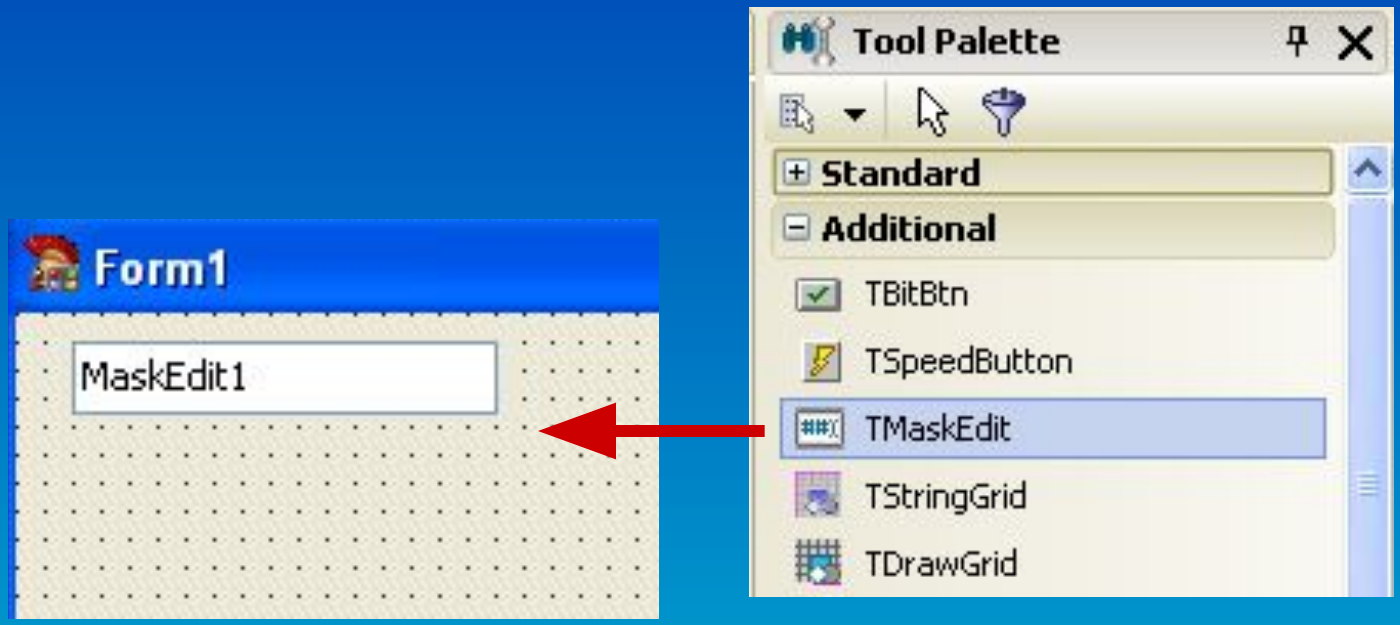
- Компонент ***GroupBox (раздел Standard)*** служит для создания ***группы*** управляющих элементов:

- Компонент ***GroupBox*** является владельцем. Это означает, что установка его свойства Visible в значение False прячет группу со всеми расположенными внутри управляющими элементами.



3. Редактор с шаблоном.

- **Поскольку управляющий элемент *Edit* не проверяет, что вводит пользователь, он неудобен для ввода данных строго определенного формата, например телефонных номеров, времени и др. На этот случай разработчики Delphi предусмотрительно поместили в Палитру Компонентов компонент *MaskEdit*:**

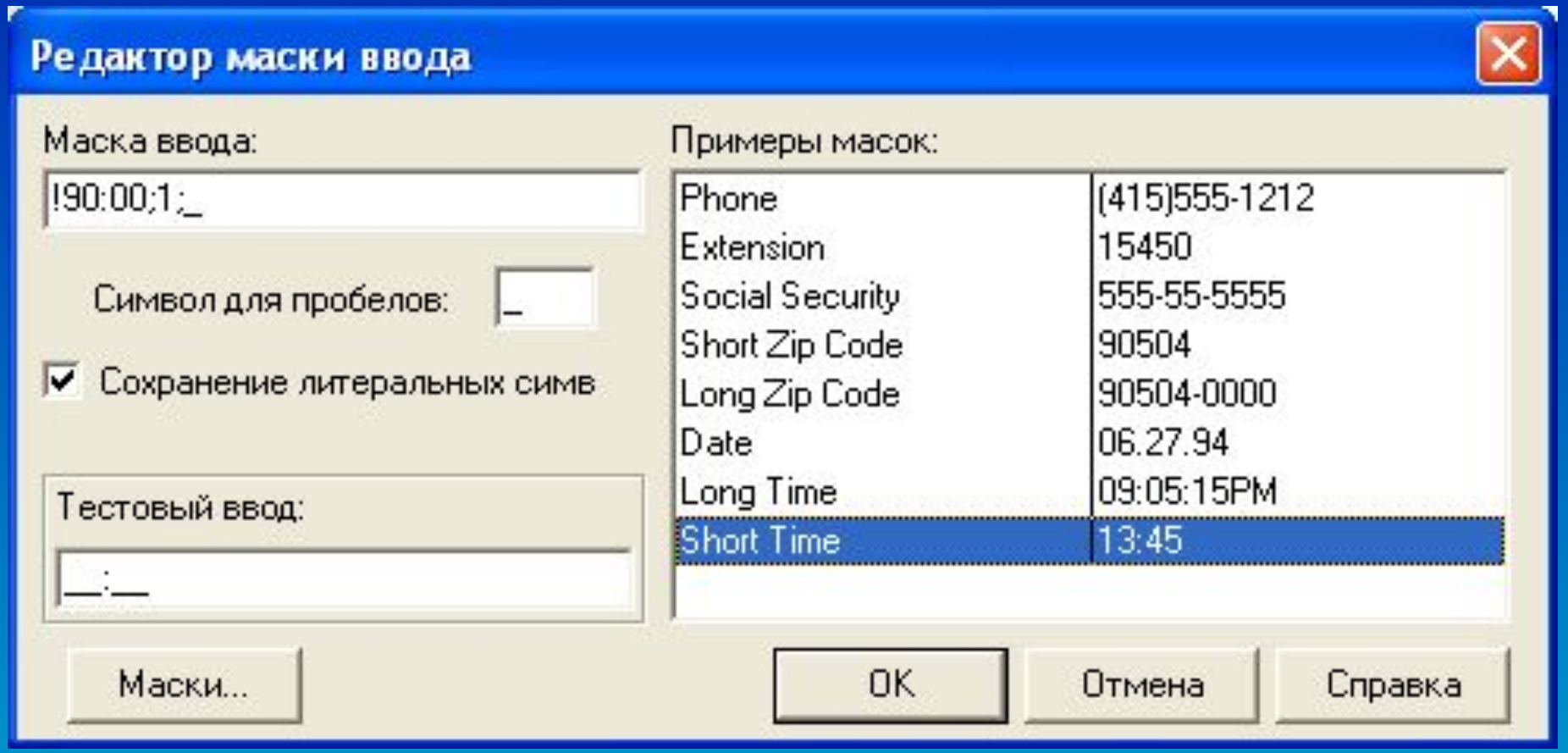


3. Редактор с шаблоном.

- **Компонент *MaskEdit* представляет собой однострочный редактор, который вынуждает пользователя вводить разрешенные символы в разрешенных позициях. Во многом аналогичный компоненту *Edit*, он отличается от последнего тем, что имеет свойство *EditMask.***
- **Свойство *EditMask* задает *шаблон (маску)* для ввода символов текста. Шаблон имеет вид текстовой строки, его символы называются форматными и управляют тем, что вводит пользователь: буквы или цифры, в каком порядке, сколько и т. д.**

3. Редактор с шаблоном.

- **Шаблон создается при помощи специального реактора, который вызывается нажатием кнопки с многоточием в поле значения свойства *EditMask:***



4. Кнопки для увеличения и уменьшения числовых значений.

- **Ввод чисел может быть значительно упрощен для пользователя при использовании пары кнопок с противоположно направленными стрелками – компонент *UpDown:***
- **Компонент *UpDown* наиболее часто применяется в сочетании с компонентом *Edit.***

4. Кнопки для увеличения и уменьшения числовых значений.

- **Характерные свойства компонента *UpDown:***

AlignButton	Положение пары кнопок относительно ассоциированного компонента: udLeft - кнопки слева, udRight - кнопки справа.
Associate	Указывает на ассоциированный компонент (обычно Edit).
ArrowKeys	Если равно True, то нажатия на клавиатуре клавиш со стрелками “вверх” и “вниз” обрабатываются так же, как и щелчки на кнопках управляющего элемента.
Thousands	Если равно True, то после каждых трех цифр десятичного числа вставляется разделитель разрядов.

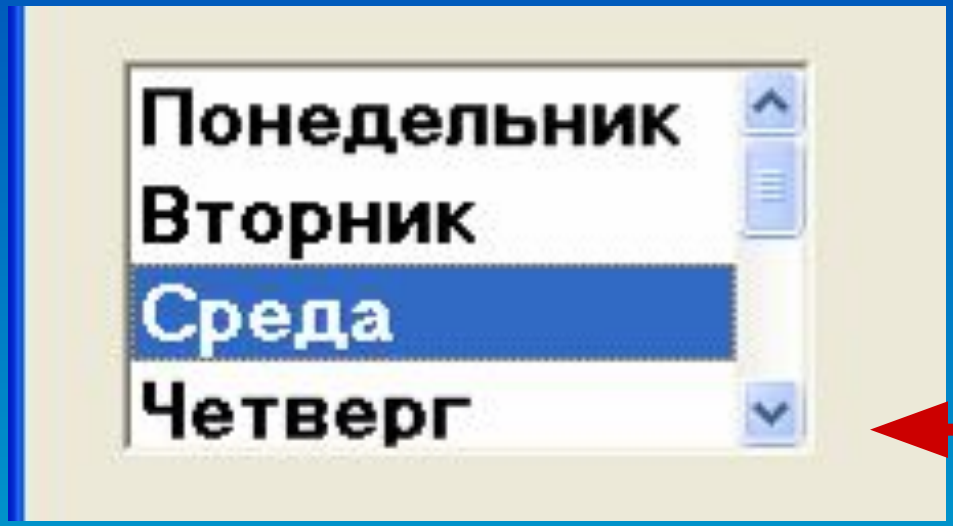
4. Кнопки для увеличения и уменьшения числовых значений.

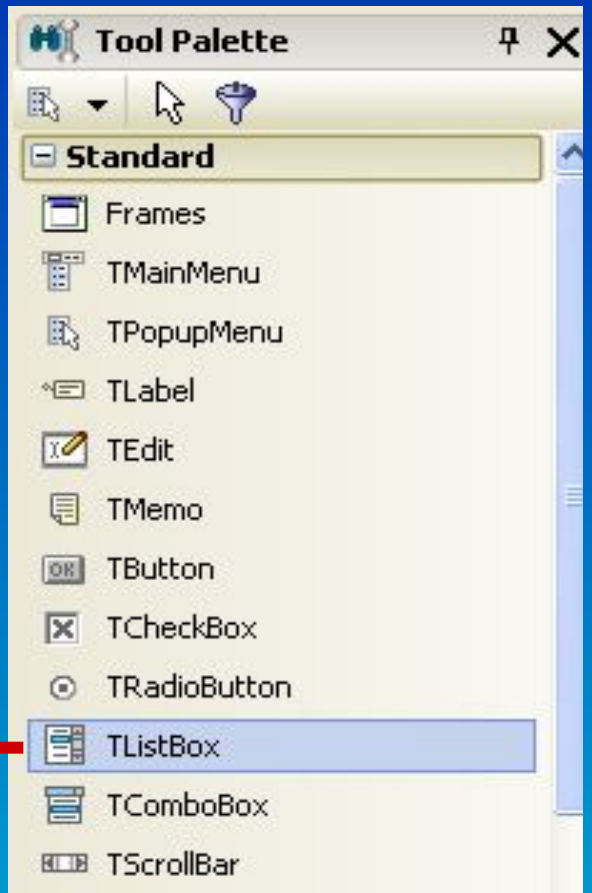
- **Характерные свойства компонента *UpDown:***

Increment	Величина, на которую увеличивается или уменьшается свойство Position в результате щелчков на кнопках со стрелками.
Min, Max	Минимальное и максимальное значения свойства Position.
Orientation	Ориентация стрелок: udHorizontal - горизонтально, udVertical -вертикально.
Position	Числовое значение, корректируемое в области ассоциированного компонента.
Wrap	Если равно True, то превышение максимального значения Max приво-дит к сбрасыванию свойства Position в минимальное значение Min.

5. Список.

- Компонент *ListBox* отображает прокручиваемый список элементов, которые пользователь может просматривать и выбирать, но не может непосредственно модифицировать.





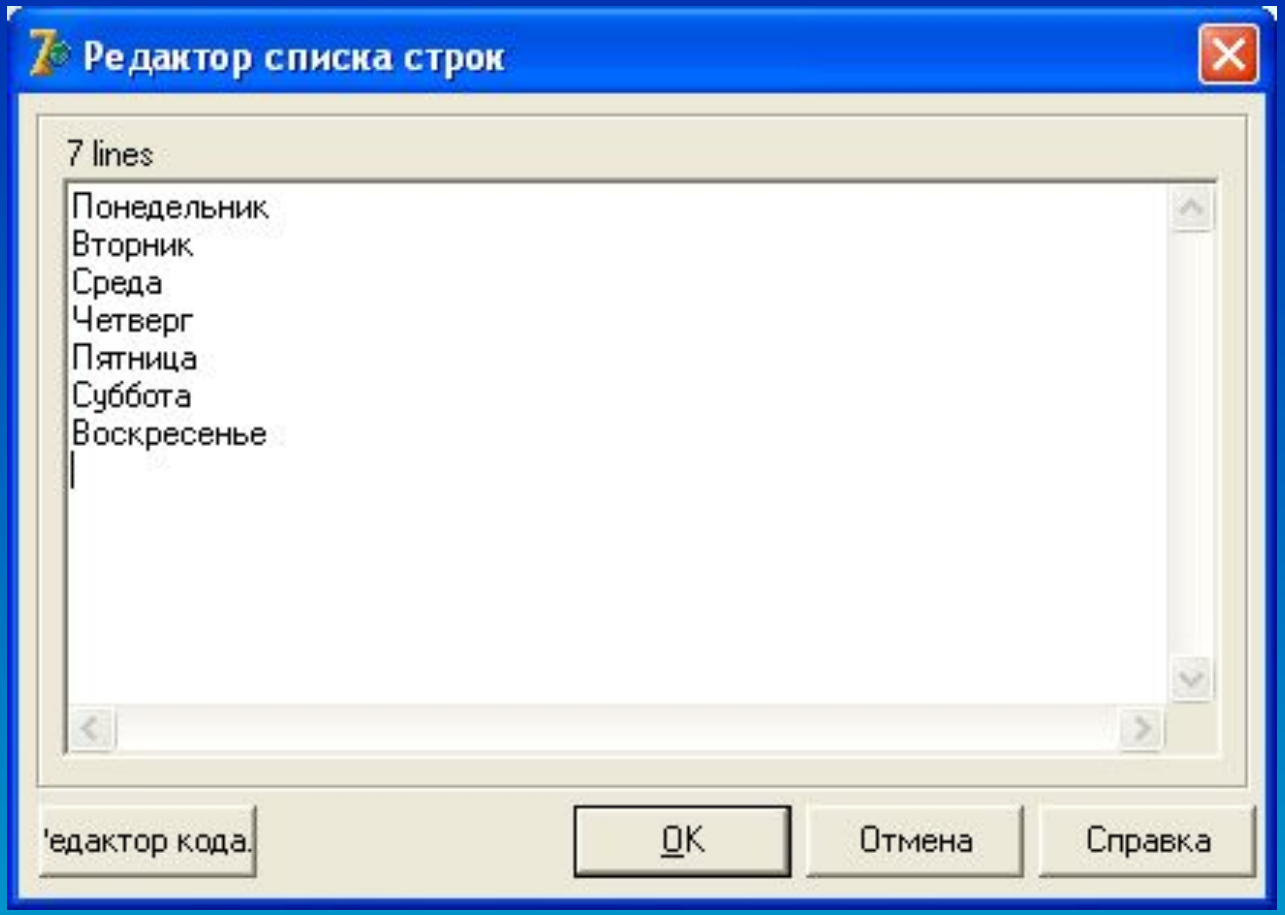
5. Список.

- **Основные свойства компонента *ListBox:***

Columns	Количество колонок в списке.
ItemHeight	Высота элемента списка, когда значение свойства Style равно IbOwnerDrawFixed.
Items	Элементы списка.
MultiSelect	Если равно True, то пользователь может выбрать в списке несколько элементов.
Sorted	Если равно True, то элементы списка сортируются в алфавитном порядке.
Style	Стиль отображения списка.

5. Список.

- Элементы списка создаются при помощи специального редактора, который вызывается при нажатии кнопки с многоточием в поле значения свойства *Items:*



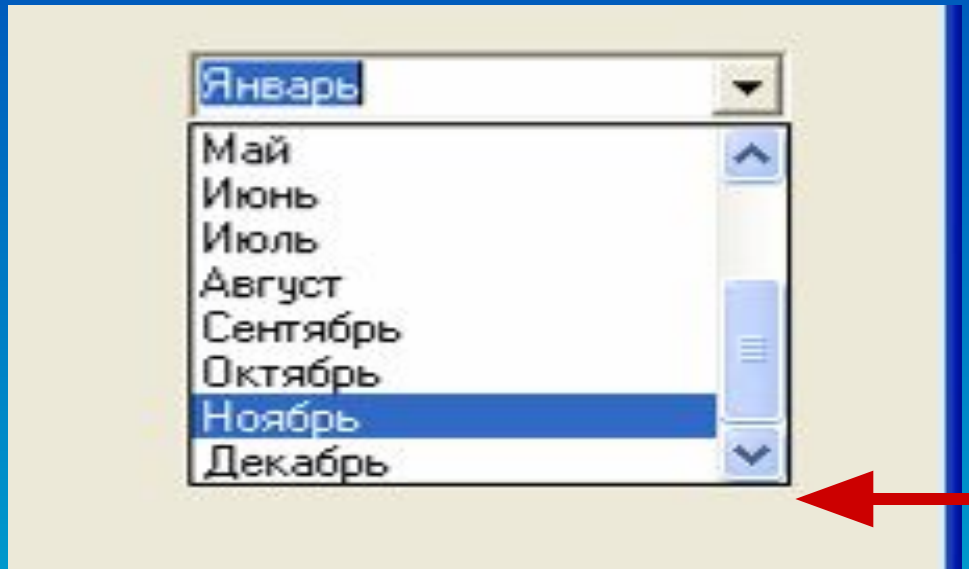
5. Список.

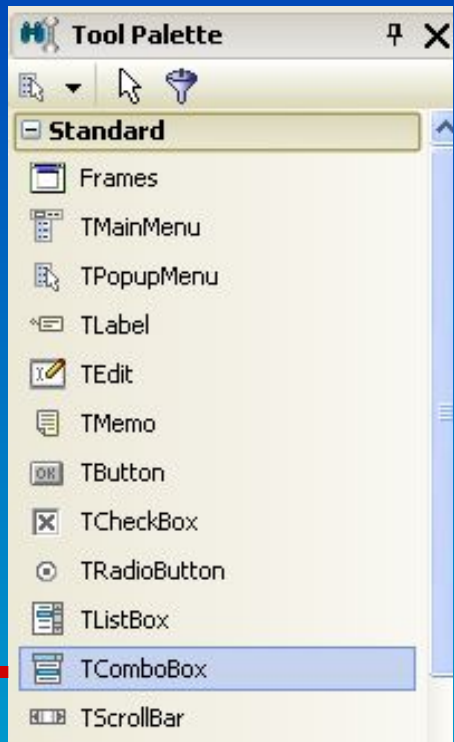
- **Нумерация элементов в *Items* начинается с нуля.**
- **Номер выбранного элемента запоминается в свойстве *ItemIndex*, доступном только программно.**
- **Пример: выбранный элемент из *ListBox1* размещается в *Edit2:***

```
Edit2.Text:=ListBox1.Items[ListBox1.ItemIndex];
```

6. Выпадающий список.

- Компонент *ComboBox* объединяет функции *ListBox* и *Edit.* Пользователь может либо ввести текст, либо выбрать его из выпадающего списка.
- Основные свойства *ComboBox* совпадают со свойствами компонентов *ListBox* и *Edit.*





6. Выпадающий список.

- **Текст выбранной или написанной пользователем строки находится в свойстве *Text.* Индекс выбранной строки можно узнать из свойства *ItemIndex.***
- **Основное событие компонента - *OnChange* наступает при изменении текста в окне редактирования в результате прямого редактирования текста или в результате выбора из списка.**
- **Пример:**

```
Edit2.Text:=ComboBox1.Text;
```

- **Далее:**

- **Лабораторная работа № 12.**
  «Проектирование окон диалога с управляющими элементами».
- **Домашнее задание (на зачет):**
  Разработать приложение «Будильник», для подачи сигналов (сообщений) в заданные моменты времени. В нём спроектировать окно диалога с управляющими элементами для создания и редактирования «будильников».

Смотри пример в учебнике «Delphi 2.0»