Лабораторная работа №1.
Создание простейшего Windows-приложения

Цели работы: Создание простейшего Windows-приложения с заданным заголовком окна и цветом формы

1. Создать папку для сохранения разработанных приложений
2. Запустить
3. Изменить заголовок окна формы с Form1 на Привет: в окне инспектора объектов (Object Inspector) установить для свойства Caption значение Привет
4. Изменить цвет формы со стандартного на другой: в окне инспектора объектов установить для свойства Color значение clAqua.
5. Выполнить приложение:

5.1. Запустить приложение - меню Run, Run или F9 или кнопка на панели инструментов.

5.2. Изменить размеры окна.

5.3. Поэкспериментировать со стандартными кнопками минимизации и максимизации окна.

5.4. Закончить работу приложения, закрыв его окно.

6. Сохранить форму и проект на диске:

Меню File, Save All, установить свою папку, создать новую папку (с именем Лабораторная работа №1), открыть ее, ввести имя проекта.

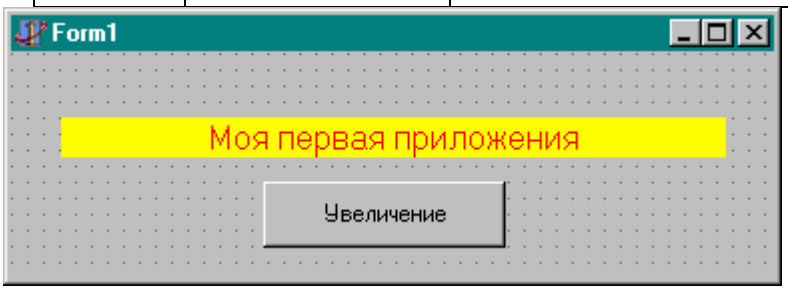
Лабораторная работа №2. Моя первая программа

Цели работы: Создание Windows-приложения, которое содержит текст "Моя первая программа!" и кнопки, позволяющие изменять размер шрифта и двигать текст

1. Поместить объект Label в окно формы Form1:
2. Переместить объект Label1 на желаемое место в форме.
3. Изменить свойства объекта Label1:

В окне инспектора объектов (Object Inspector) установить следующие значения для свойств объекта:

Объект	Свойство	Значение
Label1	Caption	Моя первая программа!
	Font	12 р., красный
	Alignment	TaCenter
	Color	Желтый (Yellow)
	AutoSize	False



4. Выполнить приложение: меню Run, Run или F9.
5. Сохранить форму и проект на диске: Меню File, Save All, установить свою папку, ввести имя Лабораторная работа №2.
6. Поместить объект Button (командная кнопка) в окно Form1. Он по умолчанию получит имя Button1. Изменить его размеры.
7. Установить свойство Caption объекта Button1 в значение "Увеличение".
8. Написать код для события Click на объекте Button1: Два раза щелкнуть по объекту Button1 в форме Между словами Begin и End написать следующий

код: Label1.Font.Size := Label1.Font.Size +2;

9. Выполнить программу. Обратить внимание на то, что происходит при нажатии кнопки с надписью "Увеличение".
10. Сохранить форму и проект на диске: Меню File, Save.
11. Создать объект "командная кнопка" для уменьшения размера шрифта в тексте.
12. Создать объект "командная кнопка" для того, чтобы двигать текст.
    Код: Label1.Left := Label1.Left + 10;
    Label1.Top := Label1.Top + 10;
13. Создать объект "командная кнопка" для того, чтобы сделать текст невидимым.
    Код: Label1.visible := false;
14. Создать объект "командная кнопка" для выхода из работы программы.
    Код: Close;
15. Сохранить форму и проект.

Лабораторная работа №3. Простейшаяматематическая программа

Цели работы: Целью работы является практическое освоение методологии и принципов создания базовых стандартных элементов интерфейса Windows-программы в среде визуального проектирования.

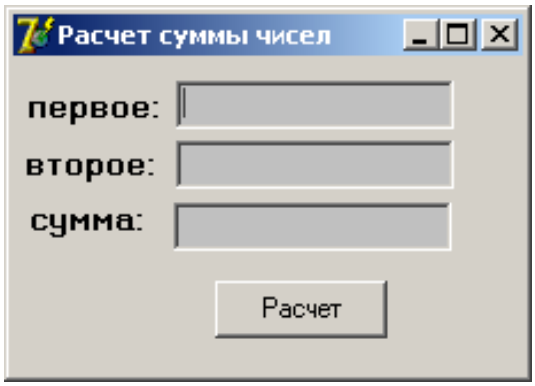
В представленном ниже проекте используем следующий минимальный набор компонент.

Button – стандартная кнопка, обычно кнопка используется для запуска действия, при этом задействуют только метод OnEvent (реакция на нажатие). Свойство Default=True ассоциирует вводимый компонент с кнопкой Enter, Cancel=True – с кнопкой Esc. Свойства Color для оформления надписи (Caption) у кнопки нет. Амперсант, помещенный в тексте надписи, указывает быструю Alt-клавишу запуска, например, Caption=A&Ppend вызывает срабатывание кнопки при нажатии Alt-P. Свойство ModalResult=true определит обязательность нажатия для закрытия дочернего окна.

Label – метка, используется как надпись или как область вывода информации для чтения. Как и для кнопки, для метки можно определить клавишу быстрого доступа, но она будет запускать связанный с меткой компонент (по FocusControl). Свойство AutoSize=True определит минимизацию размера метки под текст надписи, Aligment – центровку этого текста, WordWrap – возможность расположения текста в несколько строк, Transparent – прозрачность при наложении на другие элементы.

Edit – строка ввода. Заголовка (Caption) у этого компонента нет, но есть свойство Text как содержимое строки. Это свойство можно как считывать, так и присваивать (при необходимости с ограничением длины назначением свойства MaxLength). При вводе конфиденциальной информации указывают отображаемые символы (обычно "*"), при этом нужно переопределить свойство PasswordChar, задав его отличным от #0.

Составим проект для суммирования двух чисел, вводимых с клавиатуры.



При этом на форме нужно разместить четыре надписи (с задаваемыми свойствами Caption) и пятую надпись с пустой Caption – для отображения суммы. Определить две строки ввода для суммируемых чисел (против меток "первое" и "второе") и одну кнопку "Расчет" для запуска процедуры суммирования после ввода чисел.

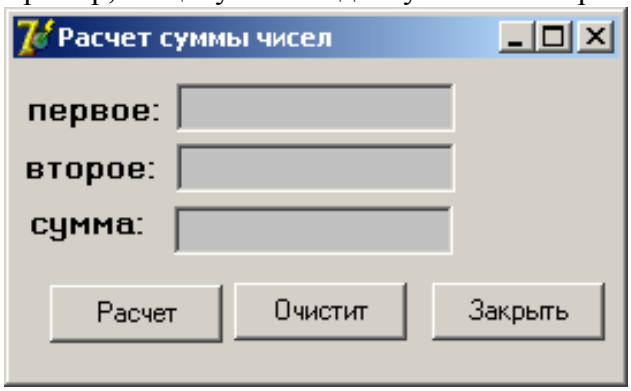
После двойного щелчка на кнопке можно заполнить шаблон процедуры реакции на нажатие этой кнопки (рамкой выделен вводимый текст).

```
procedure TForm1.Button1Click(Sender: TObject);
var a,b,c: real;
s: string; code: integer;
begin
 {ввод данных из полей редактирования}
val(edit1.text,a,code);
  val(edit2.text,b,code); c:=a+b;
str(c:–10:4,s); {перевод числа в строку}
label5.Caption:=s
end;
```

Лабораторная работа №4.
Необходимые элементы оформления проекта

Цели работы: Целью работы является практическое освоение методологии и принципов создания и оформления элементов интерфейса Windows-программы.

Приведенный выше вариант программы вполне работоспособен. Но в подобных программах обязательное требование в части их оформления – предусмотреть реакции на ввод символов в полях редактирования, например, защиту от ввода букв или второй десятичной точки. При нажатии Enter естественно переносить курсор в следующее поле редактирования или выполнять другие действия, если ввод данных завершен. В обработчиках событий (закладка Events инспектора событий) для полей ввода определим методы OnKeyPress, задав им имена, например, e1 и e2. Затем после двойного щелчка заполним шаблоны процедур.




```
procedureTForm1.e1(Sender: TObject; varKey: Char);
begin
{защита поля редактирования на ввод числа }
case key of
'0'..'9',chr(8)::
    '.': if pos('.',edit1.text)>0 then key:=chr(0);
    '–': if length( edit1.text)>0 then key:=chr(0);
  chr(13): edit2.SetFocus;
  else key:=chr(0);
 end;
end;
```

Вторая процедура отличается от первой лишь реакцией на нажатие клавиши Enter

```
procedure TForm1.e2(Sender: TObject; var Key: Char);
  begin
... edit2.text ...
        chr(13): edit2.font.color:=clRed; ...
end;
```

Текст процедуры TForm1.Button1Click желательно оформить как самостоятельную процедуру, например,

```
procedure Summa(edit1,edit2: tEdit; label5: TLabel);
```

и вызывать ее как внутри TForm1.Button1Click, так и в реакции на Enter в процедуре TForm1.e2, при этом окончание ввода данных сразу запустит вычисления.

Введем кнопку очистки полей ввода и вывода результата для нового расчета. Заголовок кнопки определим как Caption="новое", зададим реакцию OnClick (двойным щелчком на кнопке).

```
procedure TForm1.Button2Click(Sender: TObject);
   begin
  {очисткаполейввода}
   edit1.text:='';
   edit2.text:='';
   label5.caption:='';
   edit1.SetFocus
end;
```

Введемкнопкувыхода

```
procedure TForm1.Button3Click(Sender: TObject);
begin
form1.close  { завершение приложения}
 end;
```

Лабораторная работа №5.
Компоненты выбора и настройки параметров

Цели работы: Создание Windows-приложения, в котором при щелчке на радио-кнопке с названием цвета на светофоре загорается соответствующий цвет

Выбор и настройка параметров при работе с программным приложением считается стандартной частью работы пользователя с любым серьезным приложением. Это может быть как настройка самого приложения, так и определение параметров отображаемых или моделируемых в приложении процессов и явлений. Элементы интерфейса Windows-программы для основных операций такой работы в настоящее время практически стандартизированы. Рассмотрим создание этих элементов на примере работы с компонентами библиотеки VCL (Visual Component Library) в среде .

Базовые элементы выбора и настройки параметров расположены на странице Standart палитры компонент . В представленном ниже проекте используем следующий классический набор компонент:

GroupBox – группа, которая визуально и логически объединяет наборы компонент, определяет порядок перемещения по компонентам на форме (при нажатии клавиши TAB). При помещении в группу новый компонент получает свойства ParentColor, ParentShowHint, ParentFont, ParentCtl3D этой группы. Свойства Left и Top сгруппированных объектов определяются по верхнему углу группы, а не формы;

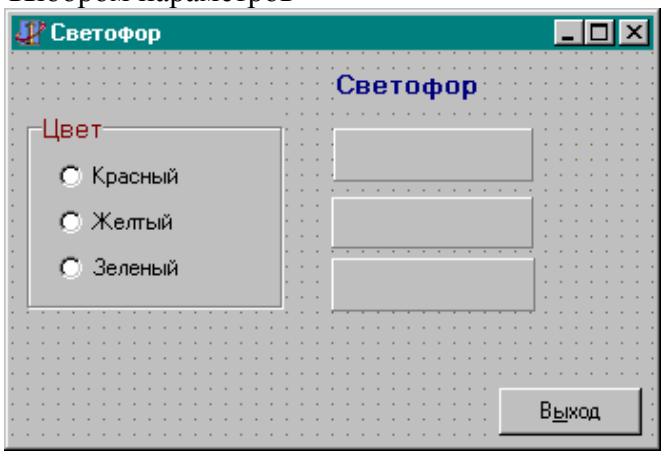
RadioGroup – группа для объектов RadioButton (см. ниже);

RadioButton – переключатели или радиокнопки, служат для выбора одной возможности из набора взаимоисключающих возможностей. Термин отражает сходство с набором кнопок выбора каналов радиоприемника. Эти кнопки обычно объединяют группой RadioGroup. Выбор кнопки отражает свойство Checked, свойство Alingment определяет положение поясняющей надписи относительно кнопки;

CheckBox – выключатель, выглядит как строка текста с окошком для установки отметки о выборе. Выключатели работают независимо, но их обычно группируют. При определении реакции на выбор можно использовать событие OnClick, но обычно устанавливают как индикатор свойство State по трем состояниям – cbChecked (есть), cbUnChecked (нет), cbGrayed (неопределенно) внутри программы. При этом для

блокировки ручного изменения этого свойства нужно установить DragMode=Automatic.

Пример проекта с выбором параметров



1. Поместить компоненты Label, Panel, GroupBox, RadioButton (страница Standard) в форму.
2. Установить следующие свойства объектов, используя Инспектор объектов:

Label1	Caption	Светофор
Panel1,2,3	Caption	
GroupBox1	Caption	Цвет
RadioButton1	Caption	Красный
RadioButton2	Caption	Желтый
RadioButton3	Caption	Зеленый

3. Записать код для процедуры обработки события Click (щелчок мыши) на объекте RadioButton1:

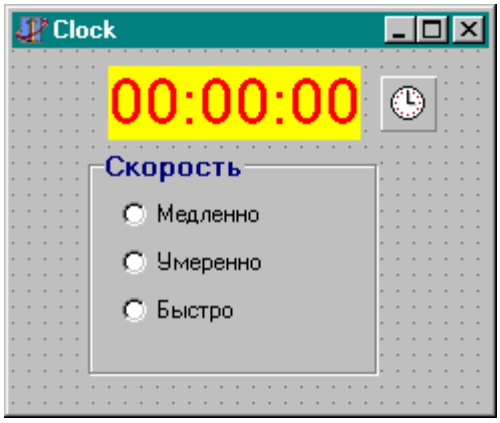
```
procedure TForm1.RadioButton1Click(Sender: TObject);
begin
        Panel1.Color := clRed;
        Panel2.Color := clWhite;
        Panel3.Color := clWhite;
end;
```

Самостоятельнозаписатькоддляпроцедур: TForm1.RadioButton2Click и TForm1.RadioButton3Click

4. Добавить печать информации "Стойте", "Внимание", "Идите" на панели с соответствующим сигналом белым цветом шрифта жирным начертанием 12п.

Лабораторная работа №6. Цифровые часы

Цели работы: Создание Windows-приложения, в котором работают цифровые часы с разной скоростью



1. Поместить компоненты Label (вкладка Standard) и Timer (System) в форму Form1.
2. Установить следующие свойства объектов

Объект	Свойство	Значение
Form1	Name	Clock
Label1	Caption	00:00:00
Label1	Color	clYellow
Label1	Font.Size	24
Label1	Font.Color	Красный

3. Записать код обновления времени для процедуры TClock.Timer1Timer: Label1.Caption:=TimeToStr(Time);
4. Добавление кнопок регулирования скорости обновления времени.

4.1. Добавить в форму компоненты **GroupBox** и **RadioButton**:

4.2. Установить следующие свойства объектов:

GroupBox1	Caption	Скорость
RadioButton1	Caption	Медленно
RadioButton2	Caption	Умеренно
RadioButton3	Caption	Быстро

4.3. Записатькоддляпроцедуры TForm1.RadioButton3Click: Timer1.Interval := 1000; Самостоятельно записать код для процедур: TForm1.RadioButton1Click (3000) и TForm1.RadioButton2Click (2000)

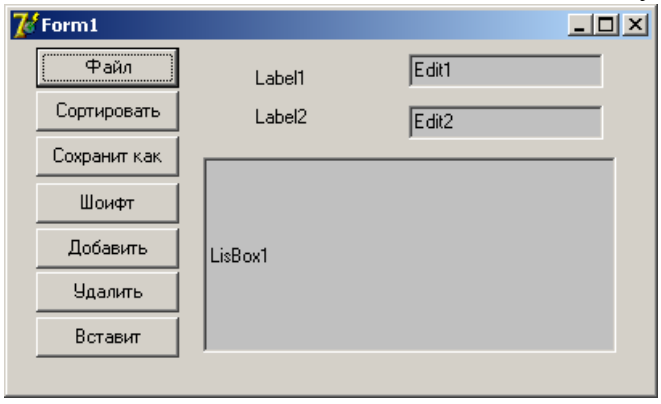
Лабораторная работа №7. Использование списков

ListBox – обычный список, этот компонент предназначен для работы с перечнем текстовых элементов (с ограничением по количеству до ~5000 шт). Перечень можно создавать (в том числе загружать как строки из текстового файла), преобразовывать и выгружать в файл. Элементы списка могут быть выбраны с помощью клавиатуры или мыши. Классический пример использования ListBox в среде Windows – выбор файла из списка в пункте меню File/Open многих приложений.

Основное свойство списка – Items (массив строк), оно аналогично свойству Lines для компонента Memo. Индекс выбранного элемента списка хранится в переменной ItemIndex. Методы Add, Delete, Insert используются для добавления, удаления и вставки строк.

Свойство Sorted=True упорядочивает список по возрастанию кода символов строк. ItemHeight – вертикальный размер элементов, Columns – число колонок в списке, ExtendedSelect – возможность множественного выбора элементов (при удержании Shift), при этом для выбранных элементов свойство Selected[номер] равно True.

ComboBox – комбинированный список, дополнительно к обычному включает строку ввода. Из нескольких типов ComboBox наиболее популярен спадающий вниз (drop-down combo box).



Создадим типовой проект с компонентом ListBox

На форме (рисунок 3).кроме списка разместим ряд кнопок (или пунктов меню), а также две строки ввода Edit1, Edit2 и две метки Label1, Label2.

По выбору пунктов организуем следующие операции со списком:

Загрузка строк из файла, имя которого предварительно набирается в строке ввода (пункт "файл")

```
Listbox1.Sorted:=false;
Listbox1.Items.LoadFromFile(Edit1.Text)
```

Сортировка списка (пункт "сортировать"):

```
Listbox1.Sorted:=true
```

Запись списка в файл, имя которого предварительно набирается в строке ввода (пункт "сохранить как"):

```
Listbox1.Items.SaveToFile(Edit2.Text);
MessageDlg('Создан файл '+Edit2.Text,mtInformation,[mbOK],0)
```

Загрузка списка экранных шрифтов (пункт "шрифт"):

```
Listbox1.Items:=Screen.Fonts
```

Добавление случайного числа в список с соблюдением сортировки, если она задана (пункт "добавить")

```
var s: string;
  begin
str(random:10:8,s);  { генерацияслучайногочисла }
ListBox1.Items.Add(s) end;
```

Добавление числа в нужное место списка (пункт "вставить")

```
var s: string;
 begin
  str(random:10:8,s);  { генерация случайного числа }
  ListBox1.Items.Insert(ListBox1.ItemIndex,s);
end;
```

Удаление выбранного элемента списка (пункт "удалить")

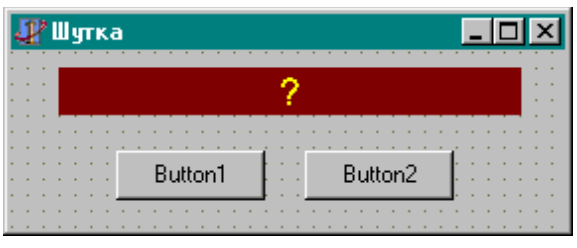
```
ListBox1.Items.Delete(ListBox1.ItemIndex)
```

Выведем некоторые характеристики выбранного элемента на метках:

```
var code: integer; a: real;
   begin
    Label2.Caption:= ListBox1.Items[ListBox1.ItemIndex];
    Val(Label2.Caption,a,code);
    If code=0 then Label1.Caption :='число'
        else Label1.Caption :='строка';
  end;
```

Лабораторная работа №8. Программа-шутка

Цели работы: Создание Windows-приложения, в котором изменяется расположения объектов.



1. Поместить компоненты Label и Button в форму в соответствии с рисунком
2. Установить следующие свойства объектов

Объект	Свойство	Значение
Form1	Caption	Шутка
Label1	Caption	?
Label1	Color	clMaroon
Label1	Font.Size	18
Label1	Font.Color	Синий
Label1	Alignment	taCenter

3. Установить свойство объекта Button2: **DragMode dmAutomatic**
4. Записать код для обработки события MouseMove на объекте Button2:

```
procedure TForm1.Button2MouseMove(Sender: TObject; Shift: TShiftState;
X,Y: Integer);
begin
        Button2.Left := Button2.Left+10;
        Button2.Top := Button2.Top+10;
end;
```

5. Записать код для обработки события Click на объекте Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
        Label1.Caption := 'Мы были в этом уверены!'
end;
```

5. Выполнить программу.
6. Развитие задачи:
Изменить программу т.о., чтобы при подводе курсора мыши к кнопке Button2 кнопка исчезала, а при отводе курсора - появлялась.

Лабораторная работа №9. Работа с окнами диалога

*Цели работы:*Целью работы является практическое освоение методологии и принципов создания элементов диалога как стандартных компонент интерфейса Windows-программы.

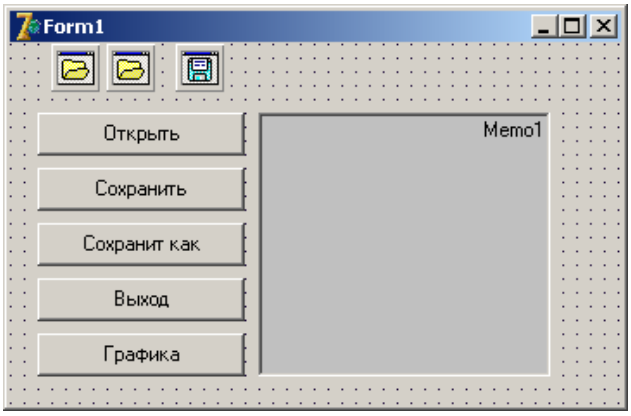
Палитра компонент содержит закладку Dialogs – диалоги работы с текстовыми и графическими файлами (открытие и сохранение), выбор цвета и шрифта, поиск и замена, работа с принтером (рисунок 4).

Объекты, представленные здесь, невидимы во время выполнения программы, – окна диалога активизируются лишь при определенных событиях, задаваемых в проекте. Чаще всего это выбор команды меню или нажатие кнопки. Характеристики и свойства диалоговых компонент приведены в приложении А.

Обычно окна диалога используются в солидных проектах с переработкой информации из файлов различных типов. Поэтому в качестве учебного примера создадим проект из двух форм – основной (Form1, свойство formStyle=fsMDIform) и дочерней (Form2, formStyle=fsMDIchild). Дочернюю форму введем из меню File/New form и затем

добавим ее в проект.

На основной форме (рисунок 5) разместим кнопки "открыть", "сохранить", "сохранить как", "выход" и "графика", а также поле Memo с Aling=alRight. Здесь же поместим три диалога – два OpenDialog (один для текста, второй – для графики) и один SaveDialog.



На дочерней форме разместим компонент Image (с закладки Additional) для вывода рисунков. Свойство Align=alClient определит заполнение по краям формы, а свойство Stretch – растяжку рисунка по границам.

Для кнопки "открыть" введем загрузку в поле примечаний содержимого файла

```
WITH OpenDialog1 Do
    If Execute Then begin
 Memo1.Visible:=True;  { видимостьполяредактора }
 Memo1.Lines.LoadFromFile(FileName);
 Caption:='Мойредактор '+
         ExtractFileName(FileName);
 SaveDialog1.Filename:=FileName;
 FileName:='';
end;
```

Длякнопки "сохранить":

```
Memo1.Lines.SaveToFile(SaveDialog1.Filename)
```

Длякнопки "сохранитькак":

```
WITHSaveDialog1 Do
If Execute Then begin
Memo1.Lines.SaveToFile(FileName);
Caption:='Мой редактор '+
         ExtractFileName(FileName);
end;
```

Для кнопки "графика" зададим деактивацию поля Memo для освобождения пространства главной формы:

```
WITH OpenDialog2 Do
     If Execute Then begin
    Memo1.Visible:=False;
    Screen.Cursor:=crHourglass; {курсор "песочные часы"}
WITH Form2.Image1.Picture  Do
     LoadFromFile(FileName);
Caption:= ExtractFileName(FileName);
Screen.Cursor:=crDefault; { нормальныйкурсор }
end;
```

Принцип использования любого стандартного окна диалога одинаков – вызывается его метод `Execute` и присваиваются возвращаемые им значения свойствам тех компонент, на которые они влияют.

Для нормальной работы диалоговых компонент необходимо определять свойство Filter (двойным щелчком в инспекторе объектов), например, для диалогов с текстовыми файлами обычно заполняют две строки "Текстовые файлы – *.txt" и "все файлы – *.*". Для графических файлов можно определить "Растры – *.bmp", "Пиктограммы – *.ico", "Метафайлы – *.wmf".

Лабораторная работа №10. Компоненты управления файлами

Цели работы: Практическое освоение методологии и принципов создания элементов управлениями файлов.

Описанные выше диалоговые панели работы с файлами общего назначения (тип OpenDialog и SaveDialog) часто несут избыточную информацию, например, когда требуется только список файлов текущего каталога или список логических устройств. В этом случае используют простые файловые компоненты с закладки Win3.1. Таких компонент здесь четыре:

FileListBox – список файлов указанного каталога;

DirectoryListBox – список каталогов указанного диска;

DriveComboBox – список логических устройств;

FilterComboBox – задание шаблона для FileListBox.

Приведем пример проекта работы с файлами с использованием простых файловых компонент.

Поместим перечисленные выше компоненты на форму вместе с компонентом Image (рисунок 6) для просмотра выбранных из списка файлов, содержащих графические изображения.

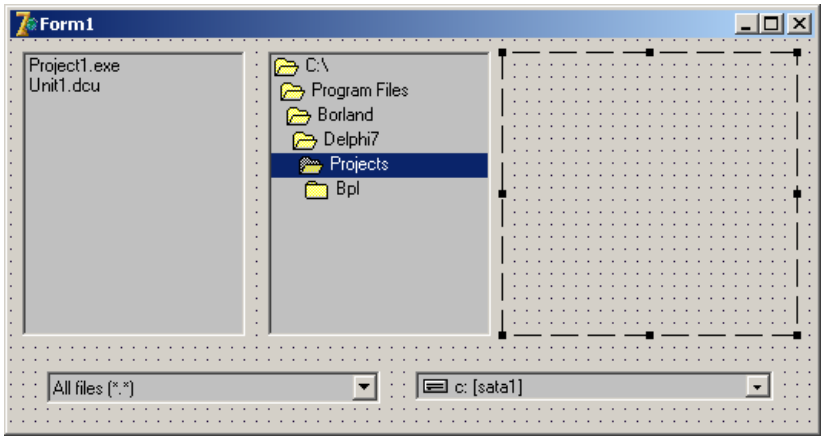


Рисунок 6

Свойство Filter шаблона файлов (FilterComboBox) определим при создании формы

```
procedure TForm1.FormCreate(Sender: TObject);
begin
FilterComboBox1.Filter:=
'метафайлы|*.wmf|'+'иконки|*.ico|'+'растры|*.bmp'
end;
```

СвойствоDblClickдляDirectoryListBoxопределяемкак

```
FileListBox1.Directory:=DirectoryListBox1.Directory
```

Свойство OnChange для DriveComboBox (сменадиска) определяемкак

```
DirectoryListBox1.Drive:=DriveComboBox1.Drive;
FileListBox1.Directory:=DirectoryListBox1.Directory
```

Выбранный файл должен отображаться в области компонента Image1, поэтому в FileListBox на событие OnClick вводим

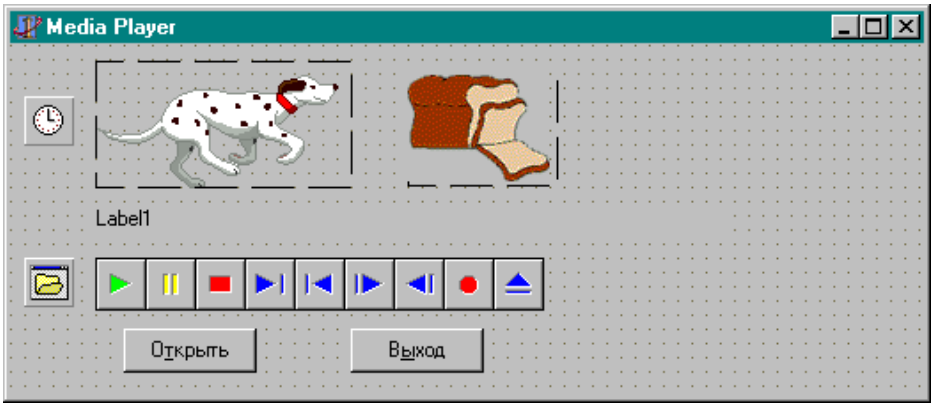
```
Image1.Picture.LoadFromFile(FileListBox1.FileName)
```

Лабораторная работа №11.
Программа с мультипликацией, видеоклипом

Цели работы: Практическое освоение методологии и принципов создания мультипликации и работа с видеоклмпами.

1. Создать в графическом редакторе Paint новый рисунок с размерами 500 на 100 пикселей. Вставив из файлов ХЛЕБА.BMP и ХЛЕБА 1.BMP рисунки, сформировать киноленту.
2. Разместить в форме панель с размерами 100x100.
5. Поместить компоненты Label, Button(вкладка Standard), OpenDialog (вкладка Dialogs) и Timer (System) в форму Form1.
3. Прямо на панель точно по левому краю разместить компонент IMAGE1 с размерами 500x100 и вставить на него киноленту.
4. Установить следующие свойства объектов

Объект	Свойство	Значение
Form1	Caption	Media Player
Label1	Caption	
Label1	Alignment	TaLeftJustify
Button1	Caption	О&ткрыть
Button2	Caption	В&ыход
OpenDialog	Filter	Аудиофайл (*.wav; *.mid)|*.wav; *.mid
RxGIFAnimator1	Animate	True



5. Поместить компоненты RxGIFAnimator(вкладка RXControls) и в форму Form1.
6. Разместить в форме таймер с интервалом 250 и для события OnTimer записать код

```
With RxGIFAnimator1 do Begin
    RxGIFAnimator1.Left:=RxGIFAnimator1.Left+10;
    Image1.Left := Image1.left + 10;
End;
```

5. Запустить и при успешной работе сохранить программу.
6. Разместить новую панель и компонент MediaPlayer1.
7. Записать код для обработки события Click на объекте Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
if OpenDialog1.Execute then begin
        MediaPlayer1.FileName:=OpenDialog1.FileName;
        Label1.Caption:= 'Загружен файл " '+ MediaPlayer1.FileName+";
        MediaPlayer1.Open;
end;
```

8. Записать код для процедуры обработки события Click (щелчок мыши) на кнопке Button2:

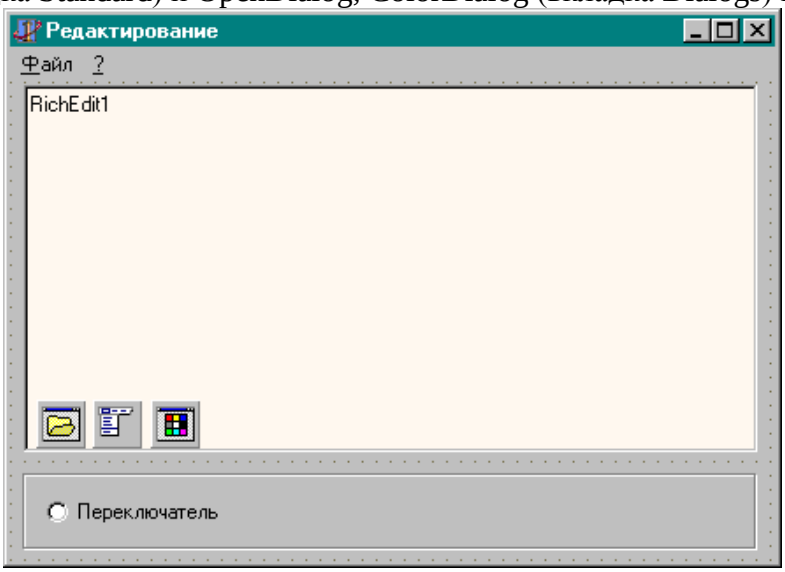
```
procedure TForm1.Button2Click(Sender: TObject);
begin
        Form1.Close;
end;
```

9. Запустить и при успешной работе сохранить программу.
9. Разместить на форме надпись с текстом гиперссылки. Установить шрифт синего цвета с подчеркиванием. При подводе курсора сделать подсказку "Сайт преподавателя" и курсор в виде ручки.

Лабораторная работа №12. Создание текстового редактора

Цели работы: Создание текстового редактора, в который можно загрузить файл, отредактировать его и оформить

1. Поместить компоненты RichEdit (вкладка Additional), MainMenu, RadioButton (вкладка Standard) и OpenDialog, ColorDialog (вкладка Dialogs) в форму Form1.



2. Вызвать текстовый редактор Блокнот и создать в нем текстовый файл My_text.txt с содержанием:

Объект	Свойство	Значение
Form1	Caption	Редактирование
RichEdit	Caption	
RichEdit	Alignment	TaLeftJustify
Открыть Ctrl+O	Caption	Открыть
-	Caption	

Выход	Caption	В&ыход
OpenDialog	Filter	Текстовые файлы (*.txt; *.rtf) |*.txt; *.rtf
RadioButton	Caption	Переключатель
GroupBox1	Caption	-

2.СохранитьфайлвпапкуRichEdit.

3. Начать новый проект и сразу сохранить его в папке RichEdit.

4. Поместить компонент RichEdit в форму и установить для свойства ScrollBars (линейки прокрутки) значение ssBorth, а для свойства Align (размещение) значение alLeft (левая часть формы).

5. Записать код для процедуры обработки события Click (щелчок мыши) на подменю OpenFile:

```
procedure TForm1.OpenFileClick(Sender: TObject);
begin
        if (OpenDialog1.Execute) then
                RichEdit1.lines.LoadFromFile(OpenDialog1.FileName);
end;
```

6. Записать код, позволяющий сохранить файл при закрытии формы:

```
procedure TForm1.CloseEditClick(Sender: TObject);
begin
        Form1.Close;
end;
```

7. Запустить программу. Добавить что-нибудь в появившийся текст. Закрыть программу.
8. Запустить ее еще раз и убедиться, что загружается откорректированный текст.
9. Запустить ее еще раз и убедиться, что сохранятся откорректированный текст.
10. Добавить группу переключателей (RadioButton) для выбора цвета окна и записать соответствующий код.
11. Добавить группу переключателей для выбора размера шрифта.
12. Добавить группу переключателей для выбора вида шрифта.

Практическое задание

1. Составить проект для нахождения корней квадратного уравнения по трем его коэффициентам, вводимым с клавиатуры. Предусмотреть реакции на некорректный ввод символов в полях редактирования (например, поставить защиту от ввода букв или второй десятичной точки). При нажатии Enter переносить курсор в следующее поле редактирования (а на завершении ввода выполнять вычисления). Предусмотреть кнопки "расчет", "новые данные" и "выход".Всем введенным компонентам задать ярлычки с оперативной подсказкой (Hints). При оформлении компонент использовать по возможности различные цвета и шрифты.
2. Выполнить проект по п.1, но с использованием меню для выполнения действий.
3. Модернизировать п.2, введя запрос пароля на вход в программу. Определить заголовок проекта (Project/Options/Application) и там же выбрать значок (Load Icon) из файла с расширением "ico". Уникальный значок можно создать в редакторе (Tools/Image Editor).
4. Составить проект "редактор текстового файла" с использованием компонента Memo. Имя загружаемого и сохраняемого файла берется из строк ввода (Edit). Предусмотреть кнопки "очистка строк ввода", "сохранить", "сохранить как" и "выход" с запросом сохранения измененного содержимого Memo.
5. Выполнить проект по п.4, но с использованием меню для выполнения действий.
6. Составить проект для визуализации выбираемого стиля, размера и цвета шрифта. Сам шрифт как набор всех латинских и русских букв (как прописных, так и строчных) отображать на метке. Каждую характеристику шрифта выбирать из набора минимум четырех радиокнопок.
7. Составить проект для анализа введенной в строке Edit информации: текстовая, числовая, прочая. В качестве индикаторов использовать набор из трех компонент CheckBox. Ввести четвертый индикатор для анализа очередного набираемого символа.

Предусмотреть кнопки "новые данные" и "выход". Всем введенным компонентам задать ярлычки с оперативной подсказкой (Hints). При оформлении компонент использовать по возможности различные цвета и шрифты.

8. Составить проект для нахождения целочисленных решений уравнения $X^2+Y^2=R^2$, то есть точек с целочисленными координатами, лежащих на окружности радиуса R. Использовать три компонента ScrollBar, первый их которых будет определять радиус в диапазоне от 5 до 25, а два других – варьировать величины X и Y от 0 до R. Величины X, Y, R, а также погрешность в решении уравнения выводить на метках. Ввести индикатор нахождения решения.
9. Составить проект для работы со списком, аналогичный описанному в разделе 4, но с использованием компонента ComboBox. При этом создать текстовый файл, содержащий минимум 20 строк, например, фамилии студентов. Отображать длину выбранного элемента списка.
10. Составить проект "редактор текстового файла" с использованием компонента ListBox. Имя загружаемого и сохраняемого файла берется из строк ввода (Edit). Предусмотреть кнопки "очистка строк ввода", "сохранить", "сохранить как" и "выход".
11. Модернизировать п.7.5, введя второй компонент ListBox для имитации двухоконного редактора файлов. Ввести также окна сообщений для подтверждения проводимых в проекте операций.
12. Составить проект с использованием окон диалога OpenDialog, SaveDialog, FontDialog, ColorDialog, FindDialog и ReplaceDialog для работы с текстовым файлом, отображающимся в поле Memo.
13. Составить проект для работы с файлами, аналогичный описанному в разделе 6, но для текстовых файлов с расширениями "pas", "txt" и "bak".

Список литературы

1. Бондарев В.М., Рублинецкий В.И., Качко Е.Г. Основы программирования. – Харьков: Фолио, 1997. – 368 с.
2. Дантеманн Д., Мишел Д., Тейлор Д. Программирование в среде / Пер. с англ. – К.: НИПФ "ДиаСофт Лтд.", 1995. –608 с.
3. Дарахвелидзе П.Г., Марков Е.П. – среда визуального программирования. – СПб.: BHV, 1996. -352 с.
4. Калверт Ч. Программирование в Windows: Освой самостоятельно за 21 день / Пер. с англ. – М.: БИНОМ, 1995. – 496 с.
5. Калверт Ч. 2. Энциклопедия пользователя / Пер. с англ..– К.: НИПФ "ДиаСофт Лтд.", 1996. – 736 с.
6. Конопка Р. Создание оригинальных компонент в среде / Пер. с англ. – К.: НИПФ "ДиаСофт Лтд.", 1996. – 512 с
7. Культин Н.Б. Программирование в Turbo Pascal 7.0 и . – СПб.: BHV, 1998. – 240 с.
8. Матчо Дж., Фолкнер Д. / Пер. с англ. – М.: БИНОМ, 1995. –464 с.
9. Матчо Дж. и др. 2. Руководство для профессионалов / Пер. с англ. – СПб.: BHV, 1997. –784 с.
10. Орлик С.В. Секреты на примерах. – М.: БИНОМ" 1996. –316 с.
11. Рубенкинг Н. Программирование в для "чайников". – К.: "Диалектика", 1996. – 304 с.
12. Сван Т. Основы программирования в для Windows 95 / Пер. с англ. – К.: "Диалектика", 1996.–480с.
13. Сурков Д.А., Сурков К.А., Вальвачев А.Н. Программирование в среде Borland Pascal для Windows. – Мн.: Высш. шк., 1996. – 432 с.
14. Федоров А. Рогаткин Дм. Borland Pascal в среде Windows. – Киев: "Диалектика", 1993. – 656 с.
15. Хендерсон К. Руководство разработчика баз данных в 2 / Пер. с англ. – К.: "Диалектика", 1996. –544с.
16. Microsoft Press. Руководство программиста по Microsoft Windows 95 / Пер. с англ. – М.: "Русская Редакция", 1997. – 600 с.