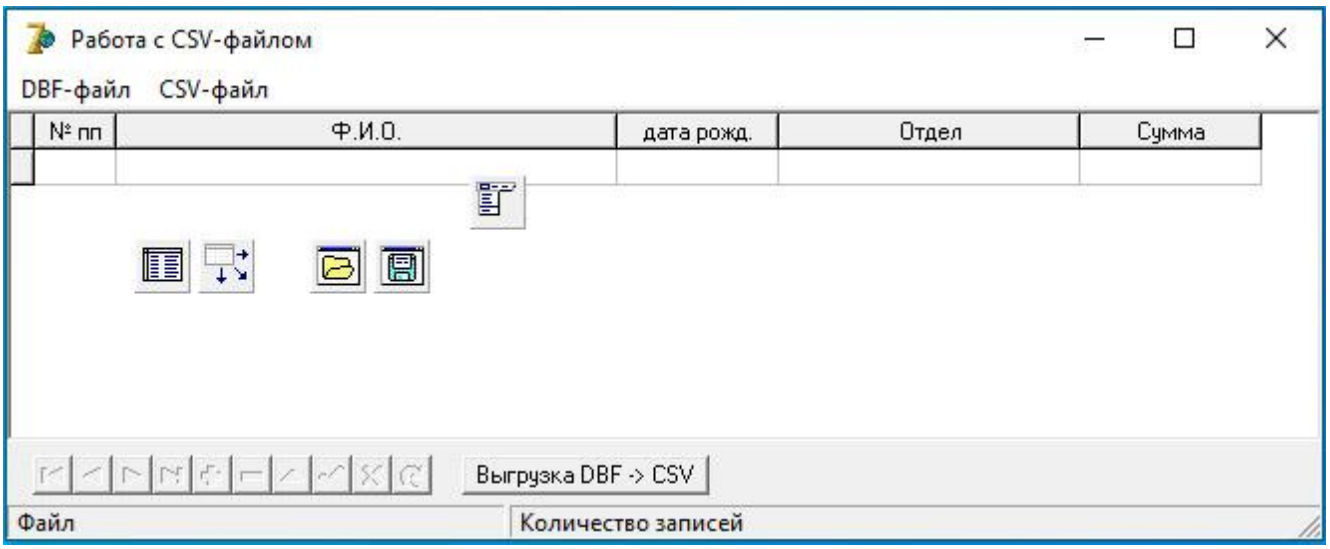
Технология компонентного программирования

Лабораторная работа № 3

Знакомство с CSV-файлом

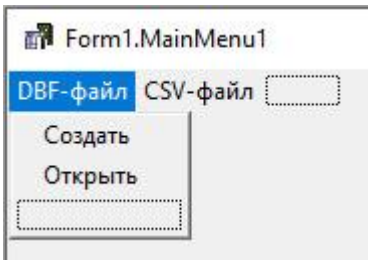
Цель занятия – знакомство с основными методами создания и чтения CSV-файла в среде Delphi.

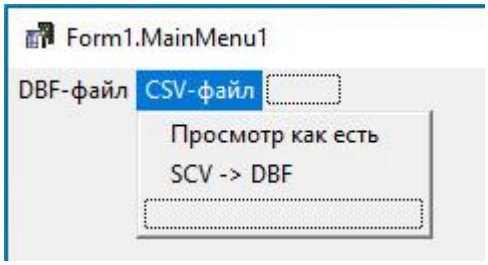
Подготовим форму следующего вида:



Здесь в нижней части формы будем использовать компонент StatusBar1, и его надо поместить на форму раньше остальных – он сам займёт низ формы. Сделаем в нём две панели.

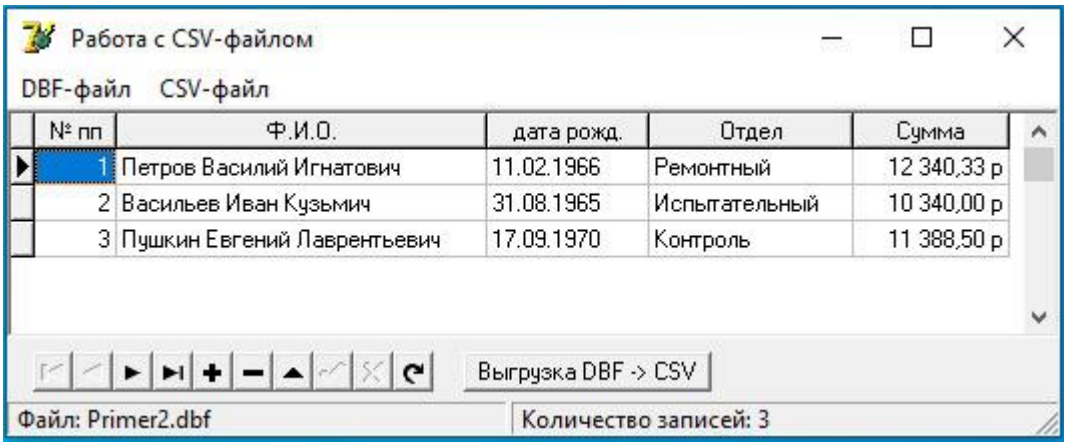
Мы здесь будем создавать DBF-файл, который будем конвертировать в CSV-файл (кнопка Выгрузка DBF -> CSV). Затем можно будет посмотреть содержимое полученного CSV-файла и переконвертировать его обратно в DBF-файл. Поэтому главное меню у нас будет иметь следующий вид:



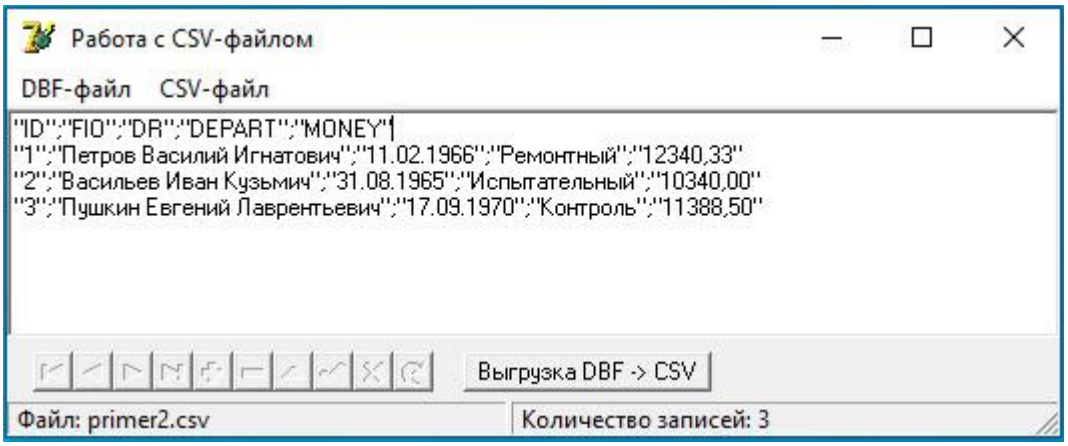


Для просмотра CSV-файла будем использовать компонент Memo1 и расположим его точно так же, как и DBGrid1, но для удобства проектирования Memo1 задвинем его под DBGrid1. На компоненте правой кнопкой мышки вызываем окно настройки, где указываем Control – Send to Back. Показывать компоненты одновременно никакой необходимости нет, поэтому такой способ вполне годится. Показывать компоненты будем через свойство Visual (True / False).

В итоге работающая программа будет иметь следующий вид:

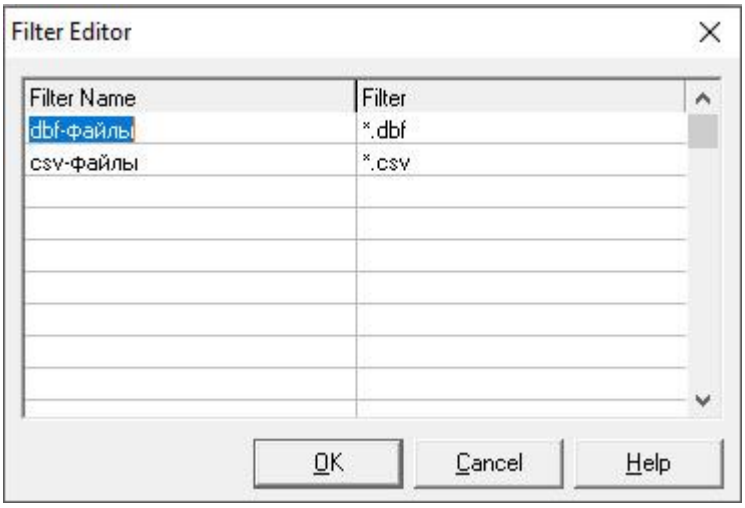


Или в режиме просмотра CSV-файла:



В работе в качестве исходного файла применим формат dBase – один из популярных форматов при обмене данными между программами разных разработчиков. В организациях часто возникает задача из своей базы сформировать данные также в формате CSV по какой-то заданной структуре. Аналогично, реальна и обратная задача – принять к себе данные в формате CSV. Поэтому в данной работе изучим форматы и DBF и CSV.

Для задания названия файла при сохранении будем использовать компонент SaveDialog1, а для загрузки готового файла – OpenDialog1. Укажем для них следующие фильтры:



Особенность применения фильтра в этих компонентах – в свойстве FilterIndex нумерация идёт с 1, а не как обычно с 0 в других компонентах.

Для удобства будем использовать папку Files для наших примеров с файлами и на неё же настроим компоненты SaveDialog1 и OpenDialog1:

```
procedure TForm1.FormCreate(Sender: TObject);
var
  WorkPath, Base: string;
begin
  WorkPath:= ExtractFilePath(Application.ExeName);
  Base:= WorkPath + 'Files';
  if not DirectoryExists(Base) then
    CreateDir(Base);
  OpenDialog1.InitialDir:= Base;
  SaveDialog1.InitialDir:= Base;
end;
```

В данном случае в качестве учебного примера создадим файл определённой структуры:

```
procedure TForm1.MnDBCreateClick(Sender: TObject);
var s,f: string;
begin
  SaveDialog1.FilterIndex:= 1;
  if not SaveDialog1.Execute then exit;
  Memo1.Visible:= false;
  DBGrid1.Visible:= true;
  f:= SaveDialog1.FileName;
  s:= UpperCase(ExtractFileExt(f));
  if s <> '.DBF' then f:= f + '.dbf';
  with TDBF.Create do begin
    ClearFields;
    AddField('ID', 'N', 5, 0);
    AddField('FIO', 'C', 60, 0);
    AddField('DR', 'D', 8, 0);
    AddField('Depart', 'C', 25, 0);
    AddField('Money', 'N', 10, 2);
    CreateTable(f, 87);
    Free
  end;
  DBFFile:= f;
  if Sender <> nil then
    ShowMessage('OK')
end;
```

Поскольку здесь используется объект TDBF, то не забудьте подключить модуль DBF3Create. Здесь задействуем глобальную переменную DBFFile, её надо внести в соответствующий блок var.

Один пункт меню есть. И второй – открытие DBF-файла:

```
procedure TForm1.MnDBOpenClick(Sender: TObject);
var f:string;
begin
  OpenDialog1.FilterIndex:= 1;
  if not OpenDialog1.Execute then exit;
  f:= OpenDialog1.FileName;
  Memo1.Visible:= false;
  DBGrid1.Visible:= true;
  Table1.Close;
  Table1.TableName:= f;
  StatusBar1.Panels[0].Text:= ' Файл: ' + ExtractFileName(f);
  Table1.Open;
  StatusBar1.Panels[1].Text:= ' Количество записей: ' +
    IntToStr(Table1.RecordCount);
end;
```

При выгрузке DBF-файла в формат CSV применяем методы работы с текстовым файлом, так как CSV-файл и есть обычный текстовый файл.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  s,v:string;
  Txt: TextFile;
begin
  if not Table1.Active then exit;
  if Table1.RecordCount = 0 then exit;
  SaveDialog1.FilterIndex:= 2;
  if not SaveDialog1.Execute then exit;
  v:= SaveDialog1.FileName;
  s:= UpperCase(ExtractFileExt(v));
  if s <> '.CSV' then v:= v + '.csv';
  Table1.First;
  AssignFile(Txt, v);
  rewrite(Txt);
  writeln(Txt,'"ID";"FIO";"DR";"DEPART";"MONEY"');
  with Table1 do
    while not Eof do begin
      s:= format('"%d";"%s";"%s";"%s";"%.2f"', [Fields[0].AsInteger,
        Fields[1].AsString, Fields[2].AsString, Fields[3].AsString,
        Fields[4].AsFloat]);
      writeln(Txt, s);
      Next
    end;
  CloseFile(Txt);
  ShowMessage('Готово')
end;
```

Загрузка CSV-файла для просмотра – также работа с текстовым файлом (чтение):

```
procedure TForm1.MnViewCSVClick(Sender: TObject);
var
  Txt:TextFile;
  f,s: string;
  k:integer;
begin
  OpenDialog1.FilterIndex:= 2;
  if not OpenDialog1.Execute then exit;
  f:= OpenDialog1.FileName;
  Memo1.Visible:= true;
  DBGrid1.Visible:= false;
  Table1.Close;
  AssignFile(Txt, f);
  reset(Txt);
  Memo1.Clear;
  k:=0;
  while not Eof(Txt) do begin
    readln(Txt,s);
    Memo1.Lines.Add(s);
    inc(k)
  end;
  CloseFile(Txt);
  StatusBar1.Panels[0].Text:= ' Файл: ' + ExtractFileName(f);
  StatusBar1.Panels[1].Text:=
     ' Количество записей: ' + IntToStr(k-1);
end;
```

Для разбора строки CSV-файла удобнее изготовить отдельную процедуру:

```
procedure Parse_csv(csv_str, sep: string; var StrList: TStringList);
var
  s,s1: string;
  k: integer;
begin
  s:= csv_str;
  StrList.Clear;
  while s <> '' do begin
    k:= pos(sep, s);
    if k > 0 then begin
      s1:= Copy(s,1,k-1);
      delete(s,1,k)
    end
    else begin
      s1:= s;
      s:= ''
    end;
    k:= length(s1);
    if k > 0 then begin
      if s1[k] = #34 then s1:= Copy(s1,1,k-1);
      if s1[1] = #34 then delete(s1,1,1);
    end;
    StrList.Add(s1)
  end;
end;
```

Тогда с применением этой процедуры для передачи CSV-файла в DBF-файл получим:

```
procedure TForm1.MnSCV_DBFClick(Sender: TObject);
var
  Lst: TStringList;
  s: string;
  k: integer;
begin
  if not Memo1.Visible then exit;
  if Memo1.Lines.Count = 0 then exit;
  MnDBCreateClick(nil);
  Table1.Close;
  Table1.TableName:= DBFFile;
  Table1.Open;
  Lst:= TStringList.Create;
  for k:=1 to Memo1.Lines.Count do begin
    s:= Memo1.Lines[k];
    Parse_csv(s, ';', Lst);
    if Lst.Count > 0 then begin
      Table1.Append;
      Table1.Fields[0].AsInteger:= StrToInt(Lst.Strings[0]);
      Table1.Fields[1].AsString:= Lst.Strings[1];   // FIO
      Table1.Fields[2].AsString:= Lst.Strings[2];   // DR
      Table1.Fields[3].AsString:= Lst.Strings[3];   // DEPART
      Table1.Fields[4].AsFloat:= StrToFloat(Lst.Strings[4]); // MONEY
      Table1.Post;
    end
  end;
  Lst.Free;
  Table1.Close;
  ShowMessage('Готово');
  Table1.Open;
  StatusBar1.Panels[0].Text:= ' Файл: ' + ExtractFileName(DBFFile);
  StatusBar1.Panels[1].Text:= ' Количество записей: ' +
    IntToStr(Table1.RecordCount);
end;
```

При закрытии формы:

```
procedure TForm1.FormClose(Sender: TObject;
   var Action: TCloseAction);
begin
  Panel1.SetFocus;
  if Table1.Modified then Table1.Post;
end;
```