INI-файлы
Теоретические основы

Цели:

- Приобрести понятие об ini-файлах
- Приобрести практические навыки использования INI-файлов
- Приобрести навыки создания кода стартовой инициализации приложения
- Приобрести навыки сохранения текущих настроек приложения

Назначение ini-файлов

Одно из главных преимуществ ini-файлов заключается в том, что эти файлы поддерживают переменные разных типов (String, Integer, Boolean). В этих файлах очень удобно хранить различные настройки, например параметры шрифта, цвет фона, какие checkbox'ы выбрал пользователь и многое другое.

Имена секций и ключей при использовании процедур записи и функций чтения должны совпадать с точностью до символа, включая написание букв (прописные и строчные)!!!

Реализация в Delphi

Для создания компонента *TIniFile* используется конструктор *Create(FileName : String),* в качестве параметра которому передается имя файла, с которым будет связан компонент. Для освобождения памяти, занятой компонентом, а так же для закрытия файла и записи файловых буферов на диск используется деструктор *Free*.

Если не вызвать деструктор Free, то возможны потери данных при записи параметров в файл!

Для чтения и записи значений у класса ***TIniFiles*** реализованы следующие методы:

- Функция ***ReadInteger(const Section: string, const Ident:string, Value: Integer): Integer*** служит для чтения числовых значений
- Функция ***ReadString(const Section: string, const Ident:string, Value: String): String*** служит для чтения строковых значений
- Функция ***ReadBool(const Section: string, const Ident:string, Value: Boolean): Boolean*** служит для чтения логических (булевых) значений
- процедура***WriteInteger(const Section: string, const Ident:string, Value: Integer)*** служит для записи числовых значений
- Процедура ***WriteString(const Section: string, const Ident:string, Value: String)*** служит для записи строковых значений
- Процедура ***WriteBool(const Section: string, const Ident:string, Value: Boolean)*** служит для чтения логических (булевых) значений

Формирование имени ini-файла

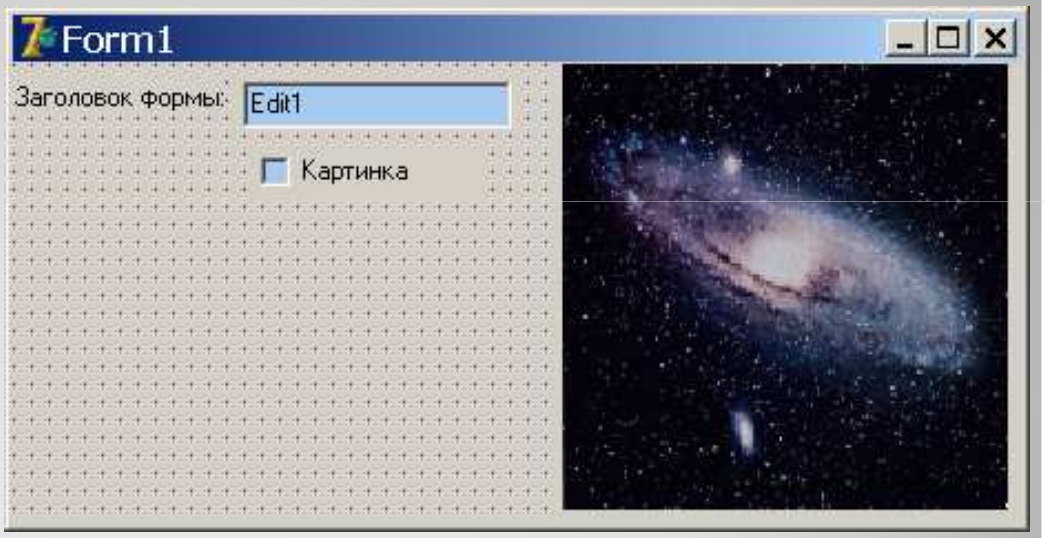
Если не задать путь к ini-файлу в конструкторе *Create*, его поиск будет осуществляться в папке Windows (на нее указывает системная переменная %SystemRoot% - обычно *C:\Windows*). Для того, чтобы поиск осуществлялся в директории программы используется свойство корневого объекта приложения *Application.ExeName*. Данное свосто содержит полный путь и имя файла выполняемого приложения (например, *C:\Delphi\Projects\Project1.exe*).

Для выделения пути из имени файла можно исплоьзовать функцию *extractfilepath(FileName : String),* которая отсекает от полного пути и имени фала имя файла (для вышеуказанного примера - *C:\Delphi\Projects*).

После того, как путь к файлу получен, остается только присоединить само имя файла (например, *MyFile.ini*).

Задание

Создадим новое приложение. Разместим на форме *TLabel, TEdit, TCheckBox и TImage*. Зададим свойства и загрузим картинку (например, из примеров Дельфи – файл *Delphi7\Demos\EarthPng\androm.bmp*). Должно получится окно, указанное на рисунке.



Добавим в секцию *uses* слово *inifiles*, а так же опишем переменную, которая будет содержать указатель на объект *in*i-файл.

```
var
 Form1: TForm1;
ini : TIniFile;
```

Зададим обработчики событий для формы. Для считывания данных перед работой приложения создадим обработчик *OnCreate* формы с учетом того, что файл должно находится в той же папке, что и приложение. В этом обработчике запишем код, производящий инициализацию размеров и заголовка формы, а так же видимость картинки.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
ini:=TIniFile.Create(extractfilepath(paramstr(0))+'
  MyFile.ini');
Edit1.Text:=ini.ReadString('FORM','Caption','Defaul
  t string');
        Width:=ini.ReadInteger('FORM','Width',
  Width);
Height:=ini.ReadInteger('FROM','Height',Height);
CheckBox1.Checked:=ini.ReadBool('IMAGE','Visible
  ',true);
ini.Free;
Image1.Visible:=CheckBox1.Checked;
Form1.Caption:=Edit1.Text;
end;
```

Если файл с таким именем существует, то он откроется для чтения, а если нет - то он будет создан. Соответственно после первого запуска программы, в ее папке появится файл *MyFile.ini*, содержащий внутри следующие строки:

```
[FORM]
Caption=Default string
Width=463
Height=237

[IMAGE]
Visible=1
```

Данный файл содержит две секции: *FROM* и *IMAGE*. В первой содержаться три ключа: *Caption, Width* и *Height*, а во второй – один логический параметр *Visisble*. В логических параметрах значение 1 соответствует истине – *true*, а 0 – *false*.

Теперь напишем обработчик события *OnClose* формы, в котором будем записывать новые значения в *ini*-файл. Код должен иметь такой вид:

```
procedure TForm1.FormClose(Sender: TObject; var Action:
  TCloseAction);
begin
ini:=TIniFile.Create(extractfilepath(Application.ExeName)
+'MyFile.ini');
 ini.WriteString('FORM','Caption',Edit1.Text);
 ini.WriteInteger('FORM','Width',Width);
 ini.WriteInteger('FORM','Height',Height);
 ini.WriteBool('IMAGE','Visible',CheckBox1.Checked);
 ini.Free;
end;
```

В этом коде все просто: открыли файл, прочитали из соответствующих секций необходимые параметры и присвоили их форме. Чтение значений из *ini*-файла по сути ничем не отличается от записи в них. Указываете секцию, где хранится необходимый параметр, указываете параметр и читаете его значение.

Дополните приложение диалогом открытия и кнопкой загрузки картинки. Добавьте ключ в *ini*-файл, который будет содержать имя загружаемой картинки и осуществлять ее автоматическую загрузку при запуске программы.