

Практическая работа №5

Теги ID3v1 файлов mp3.

Приложение «Проигрыватель с редактором ID3v1 тегов»

1. Цель работы

Освоить и закрепить на практике структуру ID3v1 тегов звуковых файлов. Применить на примере обработке тегов технологию файловых потоков чтения и записи информации двоичных файлов. Как результат, разработать прикладное приложение, позволяющее как читать информацию из ID3v1тегов, так и выполнять редактирование отдельных полей тегов.

2. Краткие теоретические сведения к практической работе

ID3 (от англ. Identify an MP3) — формат метаданных, наиболее часто используемый в звуковых файлах в формате MP3. ID3 подпись содержит данные о названии трека, альбома, имени исполнителя и т. д., которые используются мультимедиа проигрывателями и другими программами, а также аппаратными проигрывателями, для отображения информации о файле и автоматического упорядочивания аудио коллекции.

Описание тега ID3v1

Тег ID3v1 имеет строго определенную структуру, поля которой представляют собой строки, разделяющиеся нулями или пробелами. Неиспользуемые поля заполняются как пустые строки. Весь тег ID3(v1-v1.1) имеет размер в 128 байтов и располагается в конце файла после звуковых данных. Так как данный тег имеет малый объем, он присутствует во всех звуковых файлах в отличие от других версии тегов. Поскольку для данных отводилось немного места, в таких тегах можно было хранить только базовые сведения о песне: название, альбом, исполнитель, комментарий, по 30 байт на каждое поле, 4 байта для хранения года и одного байта под жанр, который можно было выбрать из заранее определённого списка из 80 значений. Если поле содержит более тридцати символов, они обрезались.

Таблица 1. Структура ID3v1

Поле	Длина	Описание
заголовок	3	«TAG»
название	30	30-символьное название
исполнитель	30	30-символьное имя исполнителя
альбом	30	30-символьное название альбома
год	4	Строковая запись года
комментарий	28[1] или 30	Комментарий
нулевой байт	1	Если номер трека присутствует, то байт равен 0.
track	1	Номер трека в альбоме или 0. Учитывается только если предыдущее поле=0.
жанр	1	Индекс в списке жанров или 255

Имеются следующие жанры:

0.Blues 1.Classic Rock 2.Country 3.Dance 4.Disco 5.Funk 6.Grunge
7.Hip-Hop 8.Jazz 9.Metal 10.New Age 11.Oldies 12.Other 13.Pop 14.R&B
15.Rap 16.Reggae 17.Rock 18.Techno 19.Industrial 20.Alternative
21.Ska 22.Death Metal 23.Pranks 24.Soundtrack 25.Euro-Techno
26.Ambient 27.Trip-Hop 28.Vocal 29.Jazz+Funk 30.Fusion 31.Trance
32.Classical 33.Instrumental 34.Acid 35.House 36.Game 37.Sound Clip
38.Gospel 39.Noise 40.AlternRock 41.Bass 42.Soul 43.Punk 44.Space
45.Meditative 46.Instrumental Pop 47.Instrumental Rock 48.Ethnic 49.Gothic
50.Darkwave 51.Techno-Industrial 52.Electronic 53.Pop-Folk 54.Eurodance
55.Dream 56.Southern Rock 57.Comedy 58.Cult 59.Gangsta 60.Top 40
61.Christian Rap 62.Pop/Funk 63.Jungle 64.Native American 65.Cabaret
66.New Wave 67.Psychadelic 68.Rave 69.Showtunes 70.Trailer 71.Lo-Fi
72.Tribal 73.Acid Punk 74.Acid Jazz 75.Polka 76.Retro 77.Musical
78.Rock & Roll 79.Hard Rock

Файловые потоки в VCL

Для чтения информации из двоичных файлов, которыми являются mp3-файлы, можно воспользоваться технологией файловых потоков.

В библиотеке VCL среды C++ Builder есть класс TFileStream – использование файлового потока (заголовочный файл Classes.hpp). Объект класса создается с помощью конструктора **new**, которому передается имя файла и режим работы с ним. Режимы могут быть:

- **fmCreate** – создается файл с указанным именем. Если файл с таким именем существовал, то он открывается для записи и все содержимое стирается.

- **fmOpenRead** – файл открывается только для чтения. Файл должен существовать.

- **fmOpenWrite** – файл открывается только для записи. Можно изменять или дополнять данные в файл, перенося текущую позицию в файле.

- **fmOpenReadWrite** – файл открывается для чтения и записи. Можно читать, редактировать и пополнять данные в файле. Файл должен существовать.

TFileStream *f;

f=new TFileStream ("My.dat", fmCreate);

После завершения работы с файловым потоком, объект надо удалить операцией **delete**.

Свойства класса TFileStream

Handle – дескриптор потока, дающий доступ к файлу (только для чтения).

Position – текущая позиция в потоке – число байтов от начала данных потока. Текущую позицию используют методы чтения-записи в поток, начиная чтение или запись именно с текущей позиции. После чтения или записи значение Position изменяется на число переданных байтов.

Size – размер данных потока в байтах (только для чтения).

f->Position=f->Size; //Перенести позицию в конец файла

Методы класса TFileStream

int Read(void *buf, int kol) – читает в буфер buf из файла kol байтов, начиная с текущей позиции (свойство Position). Метод возвращает число прочитанных байтов. Если оно меньше, чем kol, то достигнут конец файла.

int Write(void *buf, int kol) – записывает из буфера buf в поток (в файл) kol байтов, начиная с текущей позиции (свойство Position). Метод возвращает число записанных байтов. Оно может быть меньше значения kol, если не все байты удалось записать.

Seek(int dist, int dir) – изменение текущей позиции потока (файла). Параметр dist задает величину сдвига в байтах. Положительное значение – сдвиг к концу потока, отрицательное – к началу. Параметр dir указывает точку отсчета сдвига. Может принимать значения:

- **soFromBeginning** – сдвиг от начала потока (dist>=0);
- **soFromCurrent** – сдвиг относительно текущей позиции (dist любое);
- **soFromEnd** – сдвиг относительно конца потока (dist<=0).

3. Отдельные фрагменты программного кода

Для обработки ID3v1 тегов необходимо создать структуру тега.

```
struct tag //Структура тега
{
    char t[3]; //TAG
    char s1[30], s2[30], s3[30], s4[4], s5[30];
    unsigned char g; //Жанр
};
//Чтение тегов ID3v1 функциями файлового потока
TFileStream *f; //Файловый поток
tag zap; //Память для тега
//Щелчком по кнопке выполнить чтение тега и запись информации в Memo
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    unsigned char c;
    try
    {
        f=new TFileStream ("3 Раб страха.mp3", fmOpenRead);
        f->Seek(-128, soFromEnd); //На 128 к началу от конца файла
        f->Read(&zap, 128); //Читать из файла 128 байт
        Form1->Memo1->Lines->Add(zap.t); //Вывод отдельных полей
        Form1->Memo1->Lines->Add(zap.s1);
        Form1->Memo1->Lines->Add(zap.s2);
        Form1->Memo1->Lines->Add(zap.s3);
        Form1->Memo1->Lines->Add(zap.s4);
        c=zap.g;
        Form1->Memo1->Lines->Add(IntToStr(c));
    }
    catch(...)
    {
        ShowMessage("Файл отсутствует");
    }
}
```

4. Задание для выполнения практической работы

Разработать прикладное приложение «Проигрыватель с редактором ID3v1 тегов», используя технологию файловых потоков. Приложение является дальней разработкой ранее созданного приложения «Проигрыватель» при выполнении практической работы №4.

На оценку «Удовлетворительно»

При выборе песни из представленного списка звуковых файлов в компоненте ListBox, отобразить в компонентах Edit поля ID3v1 этой песни. При этом запретить пользователю редактировать поля тега.

На оценку «Хорошо»

1. Расположить кнопку «Редактировать тег». Щелчком по данной кнопке проверить права у пользователя на данную операцию путем опроса пароля.
2. После ввода правильного пароля, кнопка меняет заголовок на новый «Сохранить тег». При этом разрешается редактирование полей тега.
3. Щелчок по кнопке «Сохранить тег» выполняет сохранение введенной пользователем информации в звуковой файл.
4. Проверить правильность внесенных изменений в тег с помощью стандартного проигрывателя Windows.

На оценку «Отлично»

Осуществить поиск звуковых файлов, используя информацию из тега звукового файла. Поиск вести по автору песни и жанру.

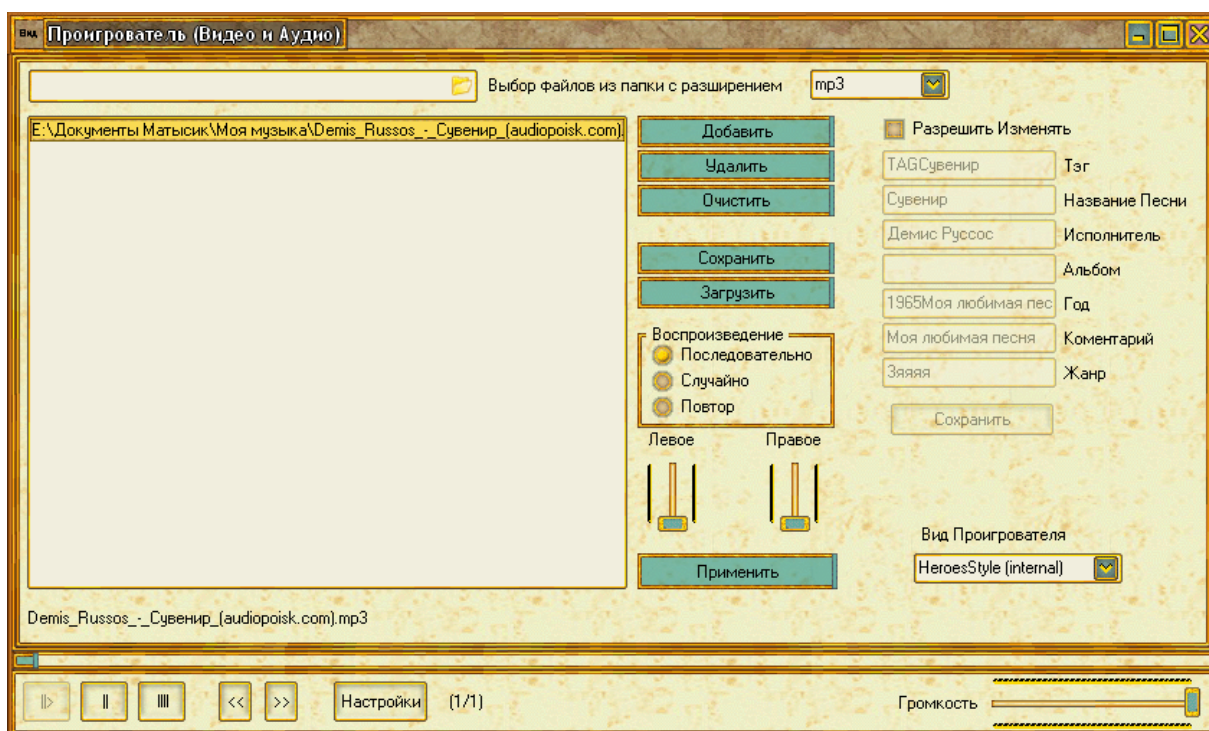


Рис. 1. Окно проигрывателя с редактором тегов