В этой статье я попытаюсь объяснить как включать файлы в приложения Delphi как различные виды ресурсов и как управлять ими.

Можно включить любой файл в исполнимый файл для использования как ресурс. Некоторые виды ресурсов признаются API и могут использоваться непостредственно. Другие просто принимаются как двоичные данные. В этой статье мы рассмотрим примеры обоих видов.

Чтобы создать файл ресурса, мы начинаем с исходного файла ***.RC**, например, по имени Resources.rc, который содержит виды ресурсов (имя, класс и файл).

sample_bmp BITMAP sample.bmp sample_ico ICON sample.ico sample_cur CURSOR sample.cur sample_ani ANICURSOR sample.ani sample_jpg JPEG sample.jpg sample_wav WAVE sample.wav sample_txt TEXT sample.txt

Имена ресурсов (*sample_bmp, sample_ico* и т.д.) произвольны. Вид ресурса может быть поддерживаемый API (**BITMAP, ICON, CURSOR**) или произвольный (**JPEG, WAVE, TEXT**). Имена файла определяют файлы, которые будут включены в **.RES** , а позже **.EXE** .

Теперь мы должны скомпилировать **.RC** файл, чтобы получился **.RES** файл. Для этого мы можем использовать Borland Resource Compiler (**brcc32.exe**).

Набрав в командной строке, мы получим:

C:DELPHIP0025>brcc32 resources Borland Resource Compiler Version 5.40 Copyright (c) 1990, 1999 Inprise Corporation. All rights reserved. C:DELPHIP0025>_

Для того, чтобы присоединить файл ресурсов к нашему проекту, мы используем директиву *{$R}* или *{$RESOURCE}*: {$R resources.res}

Загрузка поддерживаемых ресурсов (**BITMAP, CURSOR, ICON**) проста, так как Windows API предоставляет нам функции
LoadBitmap, LoadCursor, LoadIcon
соответственно для получения дескрипторов этих элементов.

```
Image1.Picture.Bitmap.Handle := LoadBitmap(hInstance, 'sample_bmp');  Icon.Handle := LoadIcon(hInstance, 'sample_ico');  Screen.Cursors[1] := LoadCursor(hInstance, 'sample_cur');
```

Другие ресурсы использовать немного сложнее. Давайте начнем с изображения **JPEG**. Мы будем использовать функцию
TResouceStream
, чтобы загрузить ресурс как поток, который будет загружен в объект
TJpegImage
.

```
function GetResourceAsJpeg(const resname: string): TJPEGImage;  var    Stream: TResourceStream;  begin    Stream := TResourceStream.Create(hInstance, ResName, 'JPEG');   try      Result := TJPEGImage.Create;      Result.LoadFromStream(Stream);    finally Stream.Free;    end;  end;
```

Пример:

```
var      Jpg: TJPEGImage;    begin      // ...      Jpg := GetResourceAsJpeg('sample_jpg'); Image2.Picture.Bitmap.Assign(Jpg);      Jpg.Free;      // ...    end;
```

Для **WAV** файлов мы нуждаемся в указателе на ресурс, загруженный в память, а для текстового файла мы должны загрузить ресурс в строку. Мы можем сделать это, используя **TResourceStream**, но давайте посмотрим пример, который использует API:

```
function GetResourceAsPointer(ResName: pchar; ResType: pchar;                                out Size: longword): pointer;  var    InfoBlock: HRSRC;    GlobalMemoryBlock: HGLOBAL;  begin InfoBlock := FindResource(hInstance, resname, restype);    if InfoBlock = 0 then      raise Exception.Create(SysErrorMessage(GetLastError));    size := SizeofResource(hInstance, InfoBlock);    if size = 0 then      raise Exception.Create(SysErrorMessage(GetLastError)); GlobalMemoryBlock := LoadResource(hInstance, InfoBlock);    if GlobalMemoryBlock = 0 then   raise Exception.Create(SysErrorMessage(GetLastError));    Result := LockResource(GlobalMemoryBlock);    if Result = nil then      raise Exception.Create(SysErrorMessage(GetLastError));  end;    function GetResourceAsString(ResName: pchar; ResType: pchar): string;  var    ResData: PChar; ResSize: Longword;  begin    ResData := GetResourceAsPointer(resname, restype, ResSize); SetString(Result, ResData, ResSize);  end;
```

Примеры вызовов:

```
var sample_wav: pointer; procedure TForm1.FormCreate(Sender: TObject); var size: longword; begin { ... } sample_wav := GetResourceAsPointer('sample_wav', 'wave', size); Memo1.Lines.Text := GetResourceAsString('sample_txt', 'text'); end;
```

Как только мы загрузим **WAV** ресурс в память, мы можем его проигрывать столько раз, сколько нужно, используя **sndPlaySound**, объявленную в модуле **MMSyste m** .

```
procedure TForm1.Button1Click(Sender: TObject); begin sndPlaySound(sample_wav, SND_MEMORY or SND_NODEFAULT or SND_ASYNC); end;
```

Есть некоторые ресурсы (типа шрифтов, анимированных курсоров), которые не могут использоваться из памяти. Мы обязательно должны сохранить эти ресурсы во временном файле на диске и загружать их оттуда. Следующая функция сохраняет ресурс в файл:

```
procedure SaveResourceAsFile(const ResName: string; ResType: pchar; const FileName: string); begin with TResourceStream.Create(hInstance, ResName, ResType) do try SaveToFile(FileName); finally Free; end; end;
```

Следующая функция использует предыдущую, чтобы сохранить ресурс во временном файле:

```
function SaveResourceAsTempFile(const ResName: string; ResType: pchar): string; begin Result := CreateTempFile; SaveResourceAsFile(ResName, ResType, Result); end;
```

Следующая функция использует **SaveResourceAsTempFile**, чтобы сохранить ресурс анимированного курсора во временном файле, затем загризить этот ресурс из файла при помощи **LoadImage** и затем удалить временный файл. Функция возвращает дескриптор, возвращенный функцией **LoadImage** .

```
function GetResourceAsAniCursor(const ResName: string): HCursor; var CursorFile: string; begin CursorFile := SaveResourceAsTempFile(ResName, 'ANICURSOR'); Result := LoadImage(0, PChar(CursorFile), IMAGE_CURSOR, 0, 0, LR_DEFAULTSIZE or LR_LOADFROMFILE); DeleteFile(CursorFile); if Result = 0 then raise Exception.Create(SysErrorMessage(GetLastError)); end;
```

Пример вызова:

```
Screen.Cursors[1] := GetResourceAsAniCursor('sample_ani');  Form1.Cursor := 1;
```