Иногда возникает необходимость поместить ресурсы в исполняемый файл Вашего приложения (например чтобы предотвратить их случайное удаление пользователем, либо, чтобы защитить их от изменений). Данный пример показывает как поместить любой файл как ресурс в *EXE*.

Далее рассмотрим, как создать файл ресурсов, содержащий копию какого-либо файла. После создания такого файла его можно легко прицепить к Вашему проекту директивой *{$R}*. Файл ресурсов, который мы будем создавать имеет следующий формат:

- заголовок
- заголовок для нашего **RCDATA** ресурса
- собственно данные - **RCDATA** ресурс

В данном примере будет показано, как сохранить в файле ресурсов только один файл, но думаю, что так же легко Вы сможете сохранить и несколько файлов.

Заголовок ресурса выглядит следующим образом:

```
TResHeader = record    DataSize: DWORD;       // размер данных    HeaderSize: DWORD;
   // размер этой записи    ResType: DWORD;        // нижнее слово = $FFFF => ordinal
ResId: DWORD;          // нижнее слово = $FFFF => ordinalM    DataVersion: DWORD;    // *
MemoryFlags: WORD;    LanguageId: WORD;      // *    Version: DWORD;        // *
Characteristics: DWORD; // *  end;
```

Поля помеченны звёздочкой Мы не будем использовать.

Приведённый код создаёт файл ресурсов и копирует его в данный файл:

```
procedure CreateResourceFile(    DataFile, ResFile: string; // имена файлов    ResID:
Integer // id ресурсов    );  var    FS, RS: TFileStream;    FileHeader, ResHeader: TResHeader;
  Padding: array [0..SizeOf(DWORD)-1] of Byte;  begin      { Открываем файл и создаем файл
ресурса }    FS := TFileStream.Create( // для чтения данных из файла    DataFile,
fmOpenRead);    RS := TFileStream.Create( // для записи файла ресурсов    ResFile,
fmCreate);      { Создаём заголовок файла ресурсов - все нули, за исключением
HeaderSize, ResType и ResID }    FillChar(FileHeader, SizeOf(FileHeader), #0);
```

FileHeader.HeaderSize := SizeOf(FileHeader); FileHeader.ResId := $0000FFFF; FileHeader.ResType := $0000FFFF; { Создаём заголовок данных для RC_DATA файла Внимание: для создания более одного ресурса необходимо повторить следующий процесс, используя каждый раз различные ID ресурсов } FillChar(ResHeader, SizeOf(ResHeader), #0); ResHeader.HeaderSize := SizeOf(ResHeader); // id ресурса - FFFF означает "не строка!" ResHeader.ResId := $0000FFFF or (ResId shl 16); // тип ресурса - RT_RCDATA (из модуля Windows) ResHeader.ResType := $0000FFFF or (WORD(RT_RCDATA) shl 16); // размер данных - есть размер файла ResHeader.DataSize := FS.Size; // Устанавливаем необходимые флаги памяти ResHeader.MemoryFlags := $0030; { Записываем заголовки в файл ресурсов } RS.WriteBuffer(FileHeader, sizeof(FileHeader)); RS.WriteBuffer(ResHeader, sizeof(ResHeader)); { Копируем файл в ресурс } RS.CopyFrom(FS, FS.Size); { Помещаем данные в DWORD - любые старые данные удалить!} if FS.Size mod SizeOf(DWORD) 0 then RS.WriteBuffer(Padding, SizeOf(DWORD) - FS.Size mod SizeOf(DWORD)); { закрываем файлы } FS.Free; RS.Free; end;

Данный код не совсем красив, и отсутствует обработка ошибок. Правильнее будет создать класс, включающий в себя данный пример.

Извлечение ресурсов из EXE

теперь рассмотрим пример, показывающий, как извлекать ресурсы из исполняемого модуля.

Вся процедура заключается в создании потока ресурса, создании файлового потока и копировании из потока ресурса в поток файла.

```
procedure ExtractToFile(Instance: THandle; ResID: Integer;  ResType, FileName: string);
var  ResStream: TResourceStream;  FileStream: TFileStream; begin  try  ResStream := TResourceStream.CreateFromID(  Instance, ResID, pChar(ResType));  try  //if FileExists(FileName) then  //DeleteFile(pChar(FileName));  FileStream := TFileStream.Create(FileName, fmCreate);  try  FileStream.CopyFrom(ResStream, 0);  finally  FileStream.Free;  end;  finally  ResStream.Free;  end;  except on E:Exception do  begin  DeleteFile(FileName);  raise;  end;  end; end;
```

Всё, что требуется, это получить **Instance** *exe*-шника или *dll* (у Вашего приложения это Application.Instance или Application.Handle, для
dll
Вам придётся получить его самостоятельно :)

ResID тот же самый **ID** , который был присвоен ресурсу **ResType**: WAVEFILE, BITMAP,

CURSOR, CUSTOM это типы ресурсов, с которыми возможно работать

FileName это имя файла, который мы хотим создать из ресурса