Review: TCompress

Squeezing A Quart Into A Pint Pot?

by Dave Jewell

There are many types of application that benefit from data compression. Suppose you're writing a database application which creates thumbnail images of bitmap files and stashes them away as BLOBs in a database. When storing these thumbnail images, you'll find you get a huge reduction in database size by compressing them before storing. A typical Windows bitmap will readily compress to something like 5% of its original size: a 95% saving in storage! This is because .BMP files have little or no built-in compression (RLE or nothing), whereas GIF, TIF and JPG files all use compression schemes which are far superior.

The same applies if you want to store free-form text inside a database. As with bitmaps, you can get very significant size reductions with compression. Binary data doesn't compress as well as graphics and text, but even here, you can typically expect to halve the amount of storage space required.

TCompress is a non-visual component that provides data compression services. It supports Delphi 1, 2 and 3, and C++Builder too. Because it's written using native Delphi code, there's no need for an external DLL, OCX, VBX or whatever. The Delphi 3 version of the product includes a package DLL which you can use if you wish.

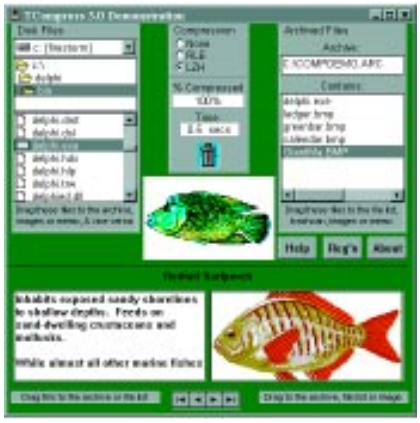
TCompress allows you to create archive files containing one or more compressed files, similar to the familiar ZIP file idea. A number of different functions are provided for creating archives, enumerating the files contained within an archive and adding/deleting files. You can also perform in-memory compression and decompression, using streams. To round things out, there are a couple of routines for working with resource data: this allows you to store compressed data as resources of type TCompress and then decompress them at run-time as required. As noted earlier, you'll find this especially beneficial if you have a lot of bitmap information to store and it has the added benefit of making your program more 'hacker-proof.'

As it comes, TCompress offers two different flavours of data compression: RLE (Run-Length Encoding) and LZH (Lempel-Ziv-Huffman). These two compression schemes have their own advantages and disadvantages. RLE will allow you to compress data very quickly, but because it's such a simple-minded scheme you won't get a very high compression factor. For instance, if you compress the Delphi 1 IDE using RLE you'll only make an 8% saving: hardly worth the bother. On the other hand, if you compress the same file with LZH, the compression will take significantly longer but you'll get a saving of around 50%. Obviously, compression ratios will be better when working with text and bitmaps.

For the Technically Curious

It should be strongly pointed out that LZH is royalty free and not to be confused with the contentious LZW (Lempel-Ziv-Welch) encoding scheme, to which Unisys holds a patent. Anyone who incorporates LZW code into a commercial product sold for profit is (as I understand it) in breach of patent law unless they apply for a licence from Unisys. In this respect, the LZH algorithm used by TCompress is 'clean'. I believe that the LZH compression scheme used by TCompress is actually the same system used by programs such as LHArc and ARJ. Specifically, a standard LZSS algorithm with an "afterburner" in the form of Huffman compression that's applied to the LZSS output stream.

A nice feature of TCompress is the ability to add your own custom



➤ *This demo is included with the TCompress evaluation version*

compression scheme. The component provides a number of On... hooks which can be used to implement custom compression algorithms. You can also add simple protection to files through the use of a 32-bit 'key' value. However, the way in which this key is implemented doesn't make for a very secure system. Rather than using the key as a seed value in the data compression, it's simply used as part of the checksum process. A better implementation might use an arbitrarily length string, as in the case of PKZIP.

For database users, the package includes three components derived from TBlobField: TCBlobField, TCMemoField and TCGraphicField. Through the use of an additional CompressSource property these components provide transparent compression/expansion support for databases.

On the negative side, the compression code in TCompress is no speed demon. On my 200MHz Pentium Pro, it takes PKZIP under 3 seconds to compress the Delphi 1 IDE: a 1.2Mb .EXE file. Compressing the same file with TCompress (LZH) takes around 14 seconds. However, as with most compression algorithms, decompression is a far simpler process and very much faster. I suspect that the poor compression may be due to the author's decision to use Delphi code throughout, rather than resorting to in-line assembler code for the critical bits.

TCompress is shareware and may be freely downloaded and used for

evaluation purposes (visit http://www.spis.co.nz). However, the evaluation copy will display a standard nag screen every once in a while. Once you register, you get a special registration number which stops the nag screen appearing. `TCompress` 3.01 has recently been released (I reviewed version 3.00) and costs about US$65 to register, with source code available at extra cost (billing is in New Zealand dollars, so US$ prices are approximate).

I cannot wholeheartedly recommend `TCompress` without some reservations. If you're planning to use it in a commercial application, I'd strongly advise you to purchase the source code because there are numerous minor bugs and irritations which could easily be fixed with access to the source. For example, the `GetAllFilesInDir` method doesn't clear the `TStringList` object before use, the `FreeFileList` routine will crash if it's passed a `Nil TStringList` and the stack usage of most methods could be drastically reduced by using `const` string parameters. Perhaps most importantly, there is plenty of room for speed improvement in the implementation of the compression code.

For further reading, one of the best books on data compression is *The Data Compression Book* by Mark Nelson, M&T Books, ISBN 0-13-202854-9.

Dave Jewell is a freelance consultant/programmer and technical journalist specialising in system-level Windows and DOS work. He is the author of *Instant Delphi Programming* published by Wrox Press. You can contact Dave as DaveJewell@msn.com, DSJewell@aol.com or DaveJewell@compuserve.com.