小杉 智昭

株式会社ミガロ.

システム事業部 プロジェクト推進室

# [Delphi/400] ファイル加工プログラミングテクニック —ファイルの圧縮・展開

略歴
1973 年 5 月 26 日
1996 年 関西大学 工学部卒
2002 年 3 月 株式会社ミガロ. 入社
2002 年 3 月 RAD 事業部配属
2007 年 4 月 システム事業部配属

現在の仕事内容
Delphi/400 を利用した受託開発とシステム保守、導入支援を担当している。



## 1.はじめに

最近はメール 1 つ取ってもプレーンテキストで書かれることは減り、文字装飾や画像・添付ファイルが加えられるといったように、何かにつけデータ容量が増大しつつある。

その半面、固定回線のような安定した高速・大容量回線から通信速度や転送量に制約のあるモバイル環境へのシフトが進み、データの圧縮・展開技術の重要度が増している。

上記を受け、本レポートでは代表的なファイルの圧縮･展開形式を確認し､ファイルの圧縮・展開といった加工を行うためのポイントをご紹介する。

## 2.代表的な圧縮･展開ファイル形式

“データ圧縮”とすると、アナログ技術を使った通信における帯域圧縮や画像・音声などに使われる非可逆圧縮も含まれるため、ここでは Windows でよく使われるアーカイブ機能（※ 1）を持ち、可逆圧縮（※ 2）を行うファイル形式に限定して取り扱うこととする。

（※ 1）アーカイブ機能

アーカイブとは書庫の意味であり、複数のファイルを 1 つのファイルにまとめることを指す。通常、アーカイブと同時にデータ圧縮が行われるが、厳密な話をするならアーカイブとデータ圧縮は別物である。

（※ 2）可逆圧縮

元のデータを損なうことなく、圧縮前のデータを完全に再現可能な圧縮方法のことを指す。画像や音声などは再現性よりも圧縮率を取ることが多く、人間があまり強く意識しない成分を無視したり、数式などで近似したりすることでデータをより圧縮する非可逆圧縮になっていることが多い。

Windows の世界で代表的なファイル圧縮・展開形式と言えば、ZIP 形式がまず挙げられる。次いで、Windows Installer などで用いられる CAB 形式が有名である。日本国内限定であれば、圧縮アルゴリズムを含め純国産の LZH 形式も非常に有名であったが、こちらは現在、開発者サイドから使用中止が呼びかけられている。これらのファイル形式について詳細を確認していく。

### ZIP形式

Windows だけでなく、コンピュータの世界で広く利用されているファイル形式で、主な拡張子は「zip」である。必要に応じて LZ77 や Deflated のような各種ある圧縮アルゴリズムを選択・使用することができるため、圧縮率重視や速度重視など、柔軟な使い分けが可能である。Windows 上での利用については、Windows98 の Plus ！パックで登場して以来、エクスプローラで標準サポートしている。

ZIP 形式にはよく似た名前の形式が多く存在しており、その代表的なものとしては、圧縮アルゴリズムが ZIP 形式と

同じ Deflate を使用している gzip や zlib である。また、7z や bzip2、rzip といったものもあるが、これらは必ずしも ZIP 形式と互換性があるわけではない。

逆に Java で使用される jar ファイルや war ファイルのように、一見すると拡張子が異なり全く違う形式のファイルに見えるものの、中身は ZIP 書庫ファイルと同等といったものも存在する。

Delphi では、以前から zlib 用のライブラリがインストールメディアに添付されていたが、これらを使って ZIP 形式のファイルとして作成するには、ファイルヘッダーやフッターといった部分を自作する必要があり、非常に手間がかかるものであった。しかし、Delphi/400 Version XE3 以降では、System.Zip ユニットが追加され、開発環境をインストールした直後から簡単に圧縮・展開プログラムを作ることが可能になった。

### CAB形式

Win32API でサポートされており、Windows Installer や ActiveX の自動ダウンロードで標準に用いられるファイル形式で、主な拡張子は「cab」である。圧縮アルゴリズムとしてはマイクロソフトが独自改良した MSZip か LZX を使用でき、ある程度の速度を保ったまま圧縮率を上げることが可能である。

この形式はマイクロソフトが開発した形式で、構造がソフトウェアの配布に向いているのが特徴でもある。過去（1998 年 11 月）に株式会社インプレスが運営するオンラインソフトウェアを紹介する Web サイト「窓の杜」で行われた 10 種類のファイル圧縮形式によるファイル圧縮対決でも優秀な成績を残している。

### 参考:LZH形式

圧縮アルゴリズムやファイル仕様など、一式全てが日本人によって開発された純国産のファイル形式で、主な拡張子は「lzh」である。圧縮アルゴリズムは LZSS 法で圧縮したデータをさらにハフマン法を用いて圧縮する LZHUF であり、lh0 から始まり lh7 方式まで公開されているが、開発途中のまま停止している。

当初、ZIP 形式の圧縮ツールが有料だったこともあり、日本国内はもとより海外でも広く使われており、国内における事実上の標準形式にまでなっていた。しかし、アンチウイルスソフトの多くが LZH 形式のファイルに対応しておらず、悪意を持って改竄された LZH 形式のファイルを検疫できない点が 2006 年にベンダーや情報処理推進機構などへ報告されたにもかかわらず、2010 年になってもベンダーの対応が行われず、いっこうに問題が解決されていない点が引き金となり、利用を控えるようにとの呼びかけが起きた。

なお、日本で圧縮ファイルを展開することを「解凍」と呼ぶのは、LZH 形式の標準ユーティリティであった LHA のマニュアルに由来する。

## 3.ファイル圧縮・展開の具体例（基礎）

まずは基礎ということで、Delphi/400 Version XE3 以降に追加された System.Zip ユニットを使ったファイルを圧縮・展開する例をご紹介する。この例では新設のユニットを利用してコード記述を行うが、難解な記述を必要としない。文章で表現すると、uses 節に System.Zip を追加し、TZipFile クラスを利用するだけである。

### TZipFileクラスのクラスメソッドを使った圧縮・展開

一番簡単な方法は、TZipFile クラスが持っている圧縮・展開を行うための専用の命令を使用することである。

ファイルを圧縮する場合には、ZipDirectoryContents メソッドを使用する。このメソッドでは、パラメータに圧縮後のファイル名と圧縮対象となるフォルダパスを指定するだけで処理できる。【コード例 3-①】

ファイルを展開する場合には、ExtractZipFile メソッドを使用する。このメソッドでは、パラメータに展開対象のファイル名と展開先のフォルダ先を指定するだけで処理できる。【コード例 3-②】

どちらのメソッドも非常にシンプルで使用方法も似ているので、簡単に圧縮・展開機能の処理を実装することができる。

### TZipFileクラスのオブジェクトを生成しての圧縮・展開

次に、TZipFile クラスからオブジェクトを生成し、操作を行う例を示す。この方法は、異なるフォルダの複数のファイルを 1 つの書庫として圧縮したり、書庫ファイル内の特定のファイルだけ展開したりするといった柔軟な処理が可能である。【コード例 3-③】

TZipFile クラスからオブジェクトを生成して利用する場合、書庫ファイルの内容にアクセスしたり、書庫にファイルを追加したり、展開したりといった作業を細かく制御できることがわかる。

上記のように TZipFile クラスを使用することで、非常に簡単に Zip ファイルを取り扱えるが、このクラスは簡易であるので、例えば暗号化ができなかったり、日本語ファイル名の対応が UTF8 のみであったりと若干使いにくい箇所もある。これらの考慮点に対応するために、オープンソースで開発されているコンポーネントを使用する方法を次章で紹介する。

## 4.ファイル圧縮・展開の具体例（応用）

基礎編では、Delphi/400 Version XE3 以降で追加された System.Zip ユニットの TZipFile クラスを使用する例を紹介したが、同時に考慮すべきこともあると述べた。記述しなかった考慮点も含めて、以下に整理する。

(1) Delphi XE 以前のバージョンに TZipFile クラスが存在しない
(2) 日本語ファイル名が UTF8 形式でしか対応していない
(3) 暗号化に対応していない（パスワード付 ZIP 書庫に対応していない）
(4) ZIP 以外のファイル形式に対応していない

これらを解決する 1 つの方法として、TurboPower Abbrevia を使う例を紹介する。TurboPower Abbrevia はオープンソースで開発されているコンポーネントで、MPL ライセンスに基づいて提供されており、以下の URL から入手可能である。

TurboPower Abbrevia
http://sourceforge.net/projects/tpabbrevia/

ここでは、2014年8月時点で最新のTurboPower Abbrevia 5.2の使用を前提とする。

上記URLからDownloadリンクを辿って入手したAbbrevia 5.2.zipファイルを展開し、IDEに組み込むと、TAbZipperやTAbUnZipper、TAbZipKitなどのコンポーネントが組み込まれる。画面上に一覧を表示したりするならTAbZipKitを、プログラム内で圧縮や展開するだけならTAbZipperやTAbUnZipperを利用するとよいだろう。

## AbbreviaをつかったZIP形式の圧縮・展開

基礎編でTZipFileクラスの専用命令を使用したのと同様の操作を、Abbreviaコンポーネントの TAbZipperとTAbUnZipperを使って行ってみる。

ファイルを圧縮する場合には、TAbZipperコンポーネントのAddFilesメソッドとCloseArchiveメソッドを使用する。このコンポーネントでは、プロパティに圧縮後のファイル名を指定し、AddFilesメソッドで圧縮対象となるフォルダパスを指定するだけで処理できる。【コード例4-①】

ファイルを展開する場合には、TAbUnZipperコンポーネントのExtractFilesメソッドを使用する。このメソッドでは、プロパティに展開対象のファイル名と展開先のフォルダ先を指定するだけで処理することができる。【コード例4-②】コード例4-①、4-②のFileName・BaseDirectoryなどのプロパティはオブジェクトインスペクタで設定可能である。

基礎編のTZipFileクラスのように簡単にZIP書庫を取り扱うことができる上に、このコンポーネントはDelphi 6～Delphi Version XE6までと非常に幅広いバージョンに対応している。また、UTF8形式以外の日本語ファイル名も対応しており、このコンポーネントを利用するだけで、前述した考慮点の(1)と(2)が解決されたことになる。(3)の暗号化も対応しており、その利用方法はTAbZipperやTAbUnZipperのPasswordプロパティをセットするだけという簡単なものになっている。コード例4－①、4－②にパスワード指定を追加した例を示す。【コード例4-①-1、コード例4-②-1】

コード例4-①-1、4-②-1のFileName・Password・BaseDirectoryなどのプロパティは、オブジェクトインスペクタで設定可能である。

残った考慮点は(4)のZIP形式以外の対応となるが、CAB形式であればTAbCabKitやTAbMakeCab、TAbCabExtractorといったコンポーネントも準備されており、圧縮・展開が可能である。

## AbbreviaをつかったCAB形式の圧縮・展開

Abbreviaコンポーネントの TAbMakeCabとTAbCabExtractorを使って、CAB形式で圧縮・展開する例を示す。基本的には使用するコンポーネントが異なるだけで、使い方はZipの圧縮・展開と同じようなコーディングで簡単に実装することができる。【コード例4-③、コード例4-④】

コード例4-③、4-④のFileName・BaseDirectoryなどのプロパティはオブジェクトインスペクタで設定可能である。

CAB形式は仕様上、暗号化に対応していないため、パスワードは指定できない。その点を除けばコンポーネントの違いはあるものの、CAB形式の書庫をZIP形式とほぼ同じ手順で取り扱うことができる。

ここまでで Abbreviaのコンポーネントの基礎的な使い方を紹介した。実は、Abbreviaのコンポーネントのうち、Zipper系のコンポーネントは、ZIP/CAB形式以外にもTAR/GZIP/BZIP2などのファイル形式にも対応しており、コンポーネントに渡すFileNameの拡張子を変更するだけで自動判別するような仕組みも持っているので、いろいろと試していただきたい。

この後は、圧縮・展開の意味からは少し外れてしまうが、自身を展開するためのプログラムが付加された実行形式の圧縮ファイルである「自己展開書庫」の作成について紹介する。

AbbreviaではTAbMakeSelfExeコンポーネントを利用することで、ZIP形式の書庫ファイルから自己展開形式のファイルを作成することが可能である。その際、事前に自己展開プログラムを準備しておく必要があるが、Abbreviaを展開した際にexamplesフォルダ内にSelfStub.dprという自己展開プログラム用のプロジェクトが用意されているので、これをコンパイルすると自己展開プログラム用の実行ファイルができ上がるようになっている。

## 参考:Abbreviaを使った自己展開書庫の作成

Abbreviaコンポーネントの TAbMakeSelfExeを使って、自己展開書庫を作成する例を示す。自己展開書庫の元となるZIP形式の書庫ファイルと、前述したexamplesフォルダ内のSelfStub.dprをコンパイルしてできるSelfStub.exeを事前に準備しておく必要がある点にご注意いただきたい。このコンポーネントも使い方は通常の圧縮・展開と同じなので、作成する自己展開書庫と対象の圧縮ファイルを指定するだけで実装できる。【コード例4-⑤】

コード例4-⑤のStubExe・ZipFileのプロパティはオブジェクトインスペクタで設定可能である。

自己展開プログラムであるSelfStub.exeを工夫することで、自己展開前にバージョン情報や注意事項を画面表示するといったことも可能である。また、出力先フォルダの指定方法などに工夫を凝らすこともできる。しかし、自己展開プログラムのサイズが大きくなると、必然的に自己展開書庫のサイズも大きくなってしまうため、必要最小限にとどめる必要がある。自己展開書庫はユーティリティプログラムではなく、自分自身の展開に特化した単機能プログラムであることを忘れてはならない。

# 5.まとめ

今回はファイルの圧縮・展開、特にZIP形式に重点を置き、その操作方法を紹介した。先に述べたようにデータの大容量化、モバイル環境へのシフトといった流れは、圧縮・展開といった技術への

コード例3-①　TZipFileクラスのクラスメソッドを使った圧縮例

```
1  procedure TfrmZIPSample.btnZipClick(Sender: TObject);
2  begin
3    // C:¥Tempフォルダの内容をC:¥test.zipに圧縮する
4    TZipFile.ZipDirectoryContents('C:¥test.zip', 'C:¥Temp¥');
5  end;
```

コード例3-②　TZipFileクラスのクラスメソッドを使った展開例

```
1  procedure TfrmZIPSample.btnUnZipClick(Sender: TObject);
2  begin
3    // C:¥test.zipの内容をC:¥Tempフォルダに展開する
4    TZipFile.ExtractZipFile('C:¥test.zip', 'C:¥Temp¥');
5  end;
```

コード例3-③　TZipFileクラスを利用する例

```
1  procedure TfrmZIPSample.btnZipClassClick(Sender: TObject);
2  var
3    zip: TZipFile;
4    i: Integer;
5  begin
6    zip := TZipFile.Create;
7    try
8      // C:¥test.zipを読み書き可能な形で開く
9      zip.Open('C:¥test.zip', zmReadWrite);
10
11     // 書庫内のファイル一覧をメモコンポーネントに列挙する
12     for i := 0 to zip.FileCount - 1 do
13       Memo1.Lines.Add(zip.FileName[i]);
14
15     // C:¥Temp¥aaa.txtを書庫に追加する
16     zip.Add('C:¥Temp¥aaa.txt');
17
18     // 書庫からbbb.txtを展開する
19     zip.Extract('bbb.txt');
20
21     // 書庫を閉じる
22     zip.Close;
23   finally
24     zip.Free;
25   end;
26 end;
```

重要度をますます強めていくことであろう。

また、通常のZIP書庫ではなく暗号化ZIP書庫を使うことで、単純にデータ容量を小さくする以外に、セキュリティを高めることも可能である点にも注目したい。

例えば、メールの送信プログラムで添付するファイルを暗号化ZIP書庫にして、メール送信後に同一宛先に対して展開パスワードを記載したメールを自動送信するようにすれば、ユーザーに手間をかけさせることなくセキュリティを高めることが可能であろう。この機会にぜひファイルの圧縮・展開といった操作を試していただければ幸いである。

なお、本レポートを作成するにあたり、zlibライブラリ、統合アーカイバコンポーネント、ZipMaster、ZipForgeといったソフトウェアを確認したが、それぞれに考慮を必要とする点があったため、参考までに以下に記しておく。

**・zlibライブラリ（Delphiのメディアに同梱または同時インストール）**

TZipFileクラス同様、新しいバージョンのDelphiでは標準でSystem.ZLibユニットとして組み込まれるようになっているため、追加インストールは不要である。また、古いバージョンのDelphiでは、インストールメディア内にライブラリ一式が同梱されており、それを使用することも、インターネットから最新ファイルを入手することも可能である。

圧縮アルゴリズムはZIP形式と互換性があるものの、ファイルヘッダーやフッターといった項目を自作する必要があり、これを使ってZIP書庫を作成したり、一般的なZIP書庫を展開したりするという目的に利用するのは難しい。しかし、ヘッダーやフッターのない専用形式として取り扱うなら、圧縮・展開機能は十分に使用可能である。

**・統合アーカイバコンポーネント**

参考：http://www.geocities.jp/norg1964/cmarc/

NIFTYサーブにて進められていた「統合アーカイバAPI仕様」に準拠した各種ファイル仕様向けのライブラリを利用するためのコンポーネントである。残念なことに開発が停止しており、対応しているDelphiのバージョンは2～7までである。他のバージョンで利用する場合は、一部修正が必要である。

**・ZipMaster**

参考：http://www.delphizip.org/

同梱されているDLLを使ってZIP書庫を作成する一風変わったコンポーネントである。開発も継続されており、XE以前の対応やXE3以降の64bit対応なども行われている。インストールが若干面倒な点と、DLLが必要となる点に注意。

**・ZipForge**

参考：http://www.componentace.com/zip_component_zip_delphi_zipforge.htm

ZIP形式の仕様をほぼ完全に準拠した高機能なコンポーネントである。公共向けや企業での利用は有償のコマーシャル版を利用する必要がある。

M

コード例4-①　TAbZipperを使った圧縮例

```
1  procedure TfrmAbZipSample.btnZipClick(Sender: TObject);
2  begin
3    // C:¥Tempフォルダの内容をC:¥test.zipに圧縮する
4    AbZipper1.FileName := 'C:¥test.zip';
5    AbZipper1.AddFiles('C:¥Temp¥*.*', 0);
6    AbZipper1.CloseArchive;
7  end;
```

コード例4-②　TAbUnZipperを使った展開例

```
1  procedure TfrmAbZipSample.btnUnZipClick(Sender: TObject);
2  begin
3    // C:¥test.zipの内容をC:¥Tempフォルダに展開する
4    AbUnZipper1.FileName := 'C:¥test.zip';
5    AbUnZipper1.BaseDirectory := 'C:¥Temp';
6    AbUnZipper1.ExtractFiles('*.*');
7  end;
```

コード例4-①-1　TAbZipperでパスワード付圧縮する例

```
1  procedure TfrmAbZipSample.btnZipClick(Sender: TObject);
2  begin
3    // C:¥Tempフォルダの内容をC:¥test.zipにパスワード付で圧縮する
4    AbZipper1.FileName := 'C:¥test.zip';
5    AbZipper1.Password := 'Password';
6    AbZipper1.AddFiles('C:¥Temp¥*.*', 0);
7    AbZipper1.CloseArchive;
8  end;
```

コード例4-②-1　TAbUnZipperでパスワードを指定して展開する例

```
1  procedure TfrmAbZipSample.btnUnZipClick(Sender: TObject);
2  begin
3    // C:¥test.zipの内容にパスワードを指定してC:¥Tempフォルダに展開する
4    AbUnZipper1.FileName := 'C:¥test.zip';
5    AbUnZipper1.Password := 'Password';
6    AbUnZipper1.BaseDirectory := 'C:¥Temp';
7    AbUnZipper1.ExtractFiles('*.*');
8  end;
```

コード例4-③　TAbMakeCabを使った圧縮例

```
1  procedure TfrmAbCabSample.btnMakeCabClick(Sender: TObject);
2  begin
3    // C:¥Tempフォルダの内容をC:¥test.cabに圧縮する
4    AbMakeCab1.FileName := 'C:¥test.zip';
5    AbMakeCab1.AddFiles('C:¥Temp¥*.*', 0);
6    AbMakeCab1.CloseArchive;
7  end;
```

コード例4-④　TAbCabExtractorを使った展開例

```
1  procedure TfrmAbCabSample.btnExtCabClick(Sender: TObject);
2  begin
3    // C:¥test.cabの内容をC:¥Tempフォルダに展開する
4    AbCabExtractor1.FileName := 'C:¥test.cab';
5    AbCabExtractor1.BaseDirectory := 'C:¥Temp';
6    AbCabExtractor1.ExtractFiles('*.*');
7  end;
```

コード例4-⑤　TAbMakeSelfExeを使った自己展開書庫の作成例

```
1  procedure TfrmAbSelfExeSample.btnMakeSelfExeClick(Sender: TObject);
2  begin
3    // C:¥SelfStub.exeをC:¥test.zipに付加して自己展開書庫を作成する
4    AbMakeSelfExe1.StubExe := 'C:¥SelfStub.exe';
5    AbMakeSelfExe1.ZipFile := 'C:¥test.zip';
6    AbMakeSelfExe1.Execute;
7  end;
```