Open Cygwin Window Here – UNIX 使いのための VBScript および C#の活用例

吉田昌英

Cygwin[1]を使っていて、画面上で開いているフォルダをカレントディレクトリとする Cygwin のシェルを起動したいと思ったことはないだろうか。

Cygwin のシェルの起動では通常、起動時のカレントディレクトリは自分の Cygwin アカウントのホームディレクトリとなる。その場合、実際に作業をするディレクトリに至るまで場合によってはかなり長い道のりを cd することになる。その代わりに、今画面上で開いているフォルダで Cygwin のシェルが開けたらかなり便利である。

この記事ではその方法を紹介する。具体的にはフォルダウィンドウ上の右クリックメニューに Cygwin のシェルを起動するための項目を加える方法を説明する。

Cygwin のシェルを操作している状態では Cygwin の世界に居るわけだが、Cygwin のシェルを起動する方法というのは Cygwin の外の世界にならざるを得ない。その結果、この記事の目的は Cygwin をより便利に使うことであるが、内容の大部分は Cygwin の外、つまり Windows に関するものにならざるを得ない。

最初私は、この記事で紹介している機能ならせいぜいスクリプトプログラミングで実現できるのではないかと想像していた。実際、VBScript を使ってある程度の機能は実現できた。しかし、最終目標に達するためにはスクリプトプログラミングでは済まず、C#で Windows Explorer の拡張モジュールを書くことになってしまった。

私は長年 UNIX 系の OS を使ってきて、最近 Windows でも主に自分の環境改善のためにプログラムを書くようになった。そういう私にとって、この記事で紹介している機能を実現するための探求およびプログラミングは、たとえて言うなら Emacs Lisp による Emacs の機能拡張を知ったときのように楽しかった。そのことから本誌の読者層にとって、Cygwin を使っているのなら、この記事の内容は十分に興味深いものだと思う。また、UNIX 使いが UNIX 系以外の環境を少し突っ込んで知るのも悪くないと思う。

この記事では副題が示すように VBScript と C#を活用する。しかし、紹介する手法をそのまま利用するだけなら VBScript や C#の知識はほとんど必要ない。またプログラムの説明の部分も深い知識は想定しておらず、必要に応じて Web ページで情報を得れば済むように書いている。また、必要なのは.NET Framework(無料)だけである。.NET Framework さえあれば C#のソースを変更してコンパイルした上で利用することもできる。

この記事で紹介する方法は Windows 2000 以降の Windows ではそのまま利用できるはずである。私自身は Windows XP SP2 と Cygwin DLL 1.5.11-1 を使っていてそれ以外の環境で検証してはいないが。

---

[1] Windows 上に X Window サーバーを含めて UNIX 的環境を提供する無料で利用できるオープンソースのプログラム群。http://www.cygwin.com

そして、この記事で紹介するプログラムは以下の URL で公開している。

http://www.geocities.jp/yoma1991/

1　課題

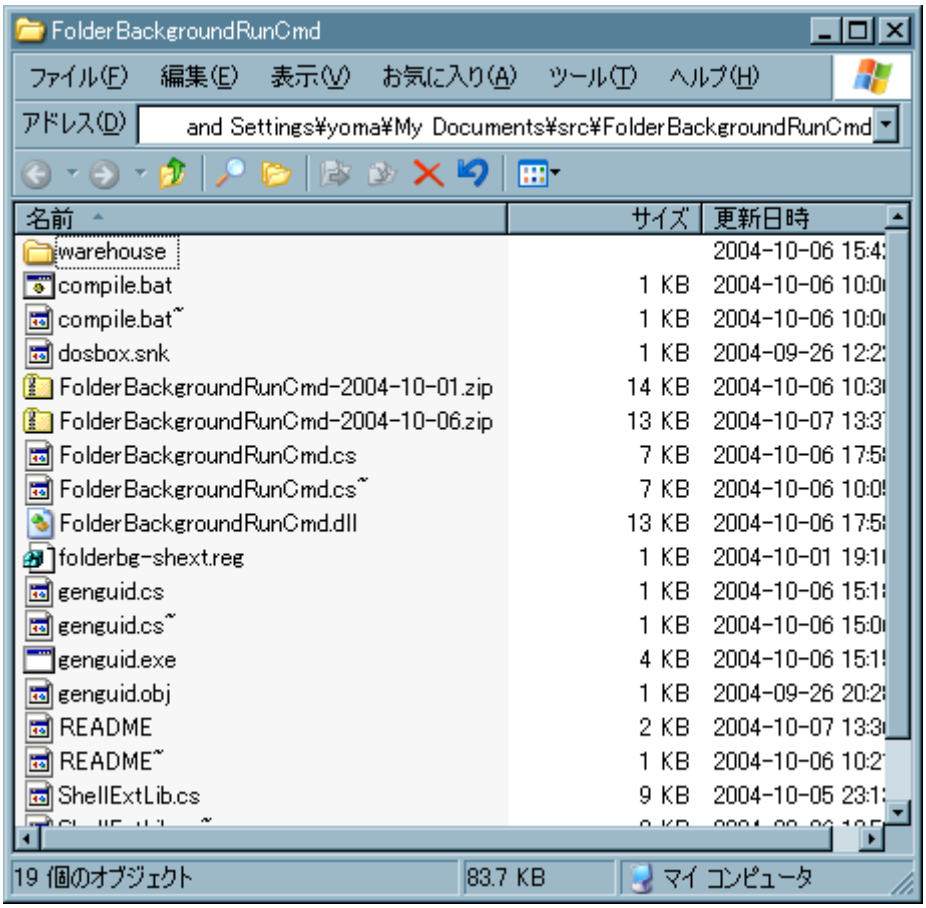
Cygwin を使っていると、画面上で開いているフォルダをカレントディレクトリとして Cygwin のシェルを動かしたいと思うことがある。たとえば図 1のフォルダウィンドウがあるとする。このフォルダのパスは以下のものだとしよう。

```
C:\Documents and Settings\yoma\My Documents\src\FolderBackgroundRunCmd
```

このフォルダをカレントディレクトリとする Cygwin のシェルを起動したいということである。

そのように感じるのは直感的に当然だと私は思うし、次のような要因もあるだろう。Windows 上でのパス名は長くなりがちで、シェルにパス名の補完機能があってもパス名の入力は面倒なことが多い。

また Cygwin の国際化機能のバグのために Cygwin 上のシェルの操作で、名前に漢字やカナを含むフォルダに cd できないこともある。文字のシフト JIS コードでの表現の 2 バイト目が 0x4C(¥)の場合、たとえばカタカナの「ソ」は注意を要する。現在の私の環境では「ソ」を含むフォルダ名も問題なく Cygwin のシェルで mkdir、rmdir、cd 等ができているが、過去に cd できないことがあったし、これから先、問題が発生しないとは限らない。Windows のレベルでカレントフォルダを移動した上で Cygwin のシェルを起動すれば少なくともフォルダに行けないということは起こらない。



[folder.png]

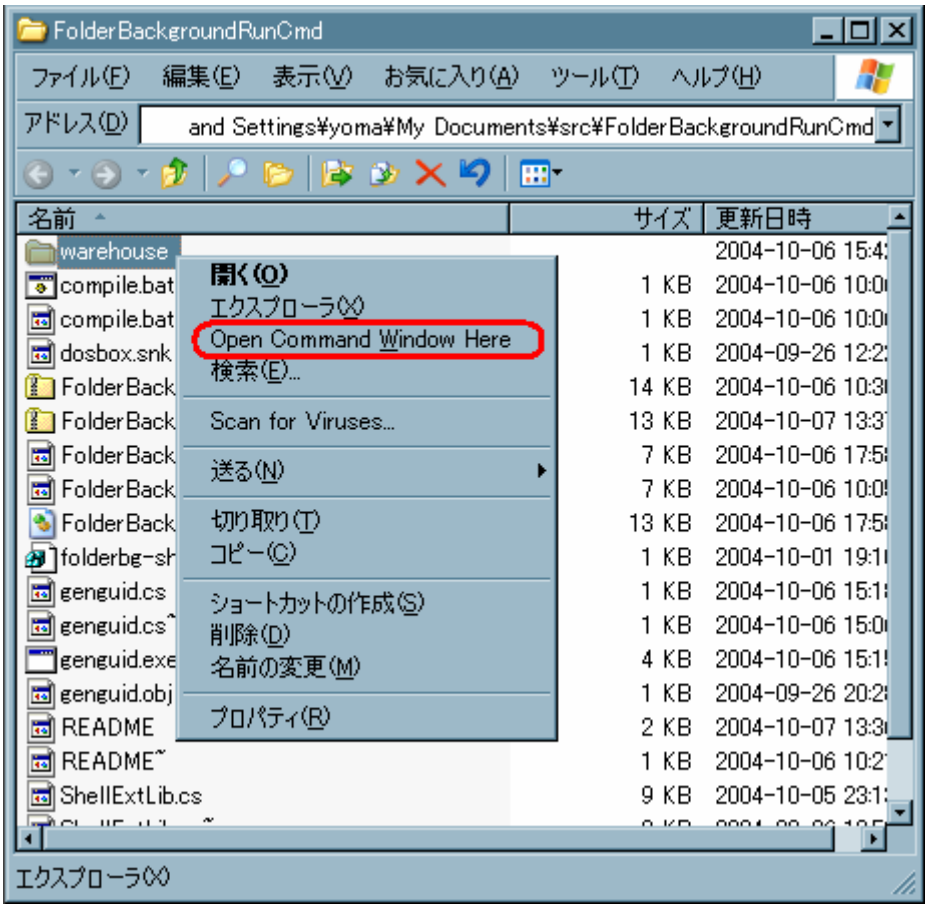
図 1

2 Open Command Window Here の利用

さて、この課題に取り組む第一歩として Open Command Window Here について考えてみよう。既に利用している人も多いと思うが PowerToys for Windows XP の 1 つに Open Command Window Here がある(以下、CmdHere と表記する)。検索サイトで「powertoys windows xp」を探せばダウンロード元はすぐにみつかる。マイクロソフトから配布されているオリジナルは英語版のみだが、日本語化されたものもある。以下の CmdHere の説明は CmdHere をインストールした上で読むほうが理解しやすいと思うが必須ではない。

オリジナルの CmdHere をインストールするとフォルダを右クリックした際に図 2のように「Open Command Window Here」というメニュー項目が現れ、それを選択すると、そのフォルダをカレントフォルダとしてコマンドプロンプトウィンドウが開く。

CmdHere の機能を使って指定したフォルダでコマンドプロンプトウィンドウを開き、その中で Cygwin のシェル、たとえば bash を動かせばある程度目標が達成できる。Cygwin の/bin ディレクトリは通常 C:¥cygwin¥bin であり、このフォルダが Windows の PATH に含まれていれば、コマンドプロンプトで単に「bash」と入力すれば bash が動作する。



[folder-cmenu-w-open-cmd-here-anno.png]

図 2

これでも何もないよりはずっといいのだが、改善の余地がある。まず、コマンドプロンプトウィンドウは使い勝手があまりよくない。具体的には以下のことが挙げられる。

- コマンドプロンプトウィンドウ上の文字をコピーしたり、コマンドプロンプトウィンドウへ文字をコピーしたりする操作がやや煩雑で、Cygwin で使える他の端末ウィンドウより劣る。
- ほとんどの端末ソフトが理解する ANSI の端末制御エスケープシーケンスの一部しかコマンドプロンプトウィンドウは理解しない。したがって、状況によっては非常に簡単に表示がおかしくなる。

また、メニューから Open Command Window Here を選んで開いたウィンドウの中に「bash」と入力するのではなく、メニューから選ぶだけで Cygwin のシェルのウィンドウが開いて欲しい。

3 Open Command Window Here のしくみ

CmdHere(Open Command Window Here)で不満な点を改善する方法を考えるに当たっては、

CmdHere のしくみを知る必要があるので、それを見ていこう。

CmdHere の実体はレジストリ登録である。フォルダウィンドウを表示しているプログラム Windows Explorer には右クリック時に表示されるメニューを拡張するしくみがある。そのしくみを使うことで CmdHere はレジストリ登録のみで実現されている。登録の内容は以下のレジストリ・キーを regedit コマンドで見れば分かる(図 3)。

HKEY_CLASSES_ROOT¥Directory¥shell¥cmd

まず、このキーの既定の値として文字列「Open Command &Window Here」が登録されている。

そして、このキーのサブキーとして command キーがり、そのキーの既定の値として文字列「cmd.exe /k "cd %L"」が登録されている。

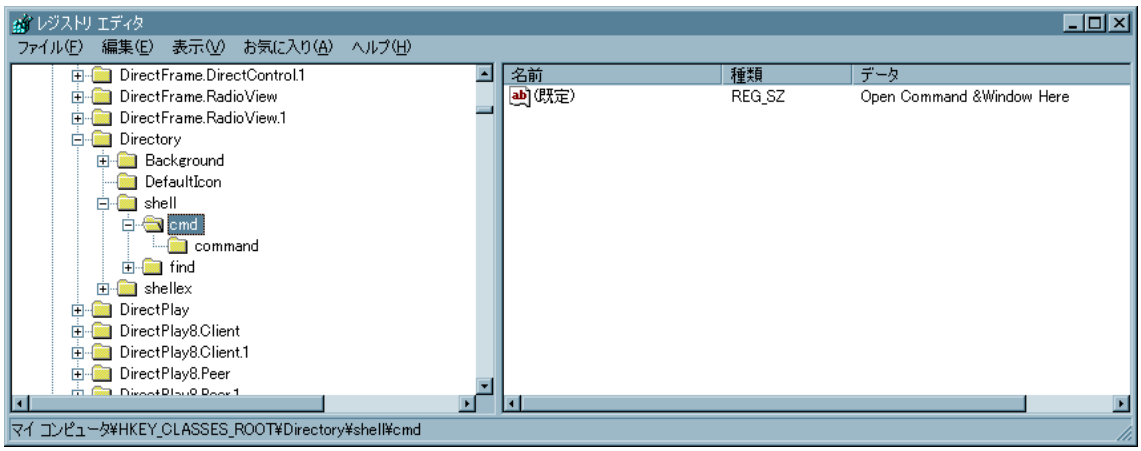
言うまでもなく cmd キーの既定の値「Open Command &Window Here」がフォルダを右クリックしたときにメニューに表示される文字列を定義している。文字列中に「&」があると、その次の文字がショートカット・キーとなる。ここでは「W」の前に「&」があるので「w」がショートカット・キーであり、「Open Command Window Here」がメニュー項目として表示されている状態で、その項目をマウスで選択するほか、「w」を押すことでもコマンドプロンプトウィンドウを開くことができる。なお、cmd というキーの名前にはあまり意味はない。

また、command キーの既定の値「cmd.exe /k "cd %L"」が、そのメニュー項目が選択された場合に実行されるコマンド行を定義している。「%L」は選択されたフォルダのパス名、この例では以下のパス名に置き換えられてからコマンド行が実行される。

```
C:\Documents and Settings\yoma\My Documents\src\FolderBackgroundRunCmd\wareh
ouse
```

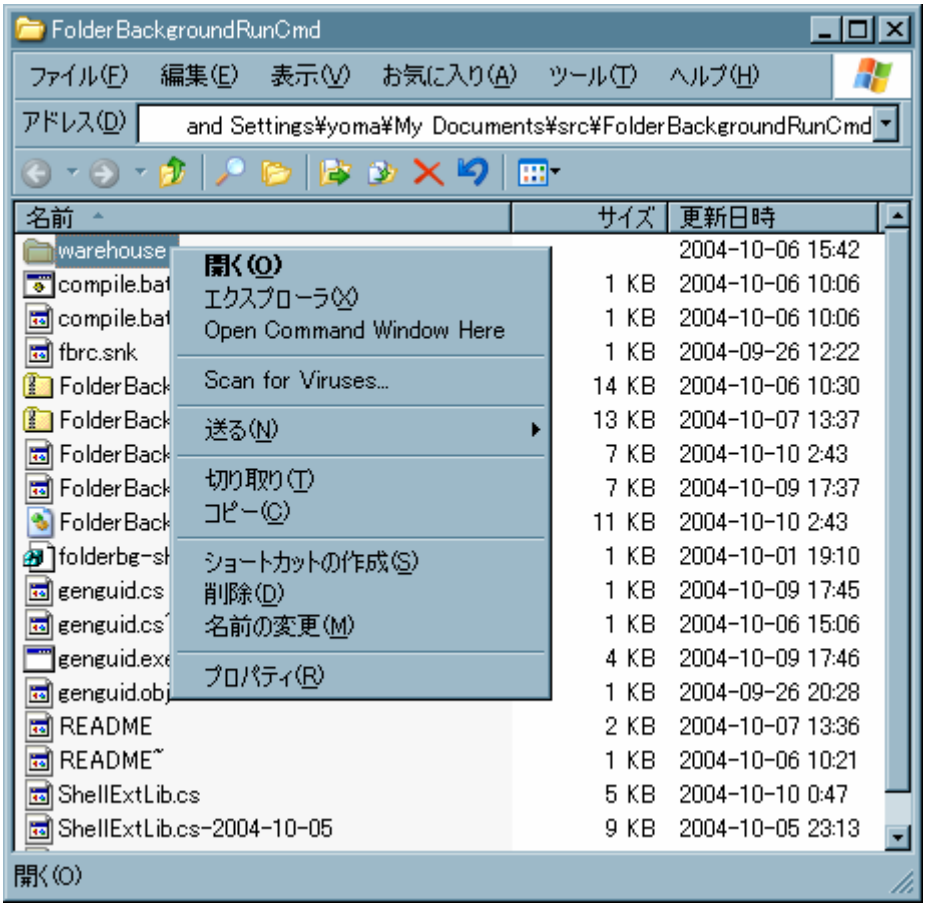
CmdHere がインストールされた状態では HKEY_CLASSES_ROOT¥Directory¥shell の下にはサブキーとして cmd のほかに find が登録されている(図 3)。find キーはフォルダを右クリックした際のメニューの中の「検索(E)...」を登録しているようだ。なぜなら find キーを削除してフォルダを右クリックすると、図 4のように「検索(E)...」がメニューに表示されない。

CmdHere 自体は Windows XP 専用であるが、ここまでで説明した CmdHere が動作するしくみは Windows 2000 以降では共通している。そして、CmdHere が cmd.exe を起動するのと同様に、Cygwin のシェルを起動するようなレジストリ登録をすれば目標が達成できそうだ。



dir-shell-cmd-reg.png

図 3



[folder-cmenu-wo-find.png]

図 4

4 Cygwin のシェルを動かすウィンドウ

フォルダを右クリックして出てくるメニューに Cygwin のシェルが動くウィンドウを開くメニュー項目を登録する方法については概ね目処がたった。次に、どんなプログラムのウィンドウの中でシェルを動かすかを考えよう。Cygwin のシェルのウィンドウでは漢字やカナの表示や入力ができることが望ましい。なぜなら日本語版 Windows では漢字やカナを使ったファイル名やフォルダ名が意図しなくても使われてしまうことがあるからである。たとえばデスクトップ・フォルダのパスは以下のとおりであり、カタカナが含まれている。

¥Documents and Settings¥*USER_NAME*¥デスクトップ

コマンドプロンプトウィンドウは漢字表示や入力に問題はないが、先に述べたように機能的に不満が残る。漢字が表示できて、その上で Cygwin のシェルが動かせるプログラムとしては以下

がある。

- kterm
- 漢字パッチを当てた rxvt
- CygTerm

私はこの中の CygTerm を使っている。CygTerm は Tera Term の中で Cygwin のシェルを動かすプログラムで、検索サイトで「CygTerm」を探せばソースコードがすぐみつかだろう。。

私の PC では CygTerm の実行形式が以下のパスに保存されている。

C:¥Program Files¥cygterm¥cygterm.exe

以下の説明ではフォルダに対する右クリックでこのパスの cygterm.exe を起動することをめざす。

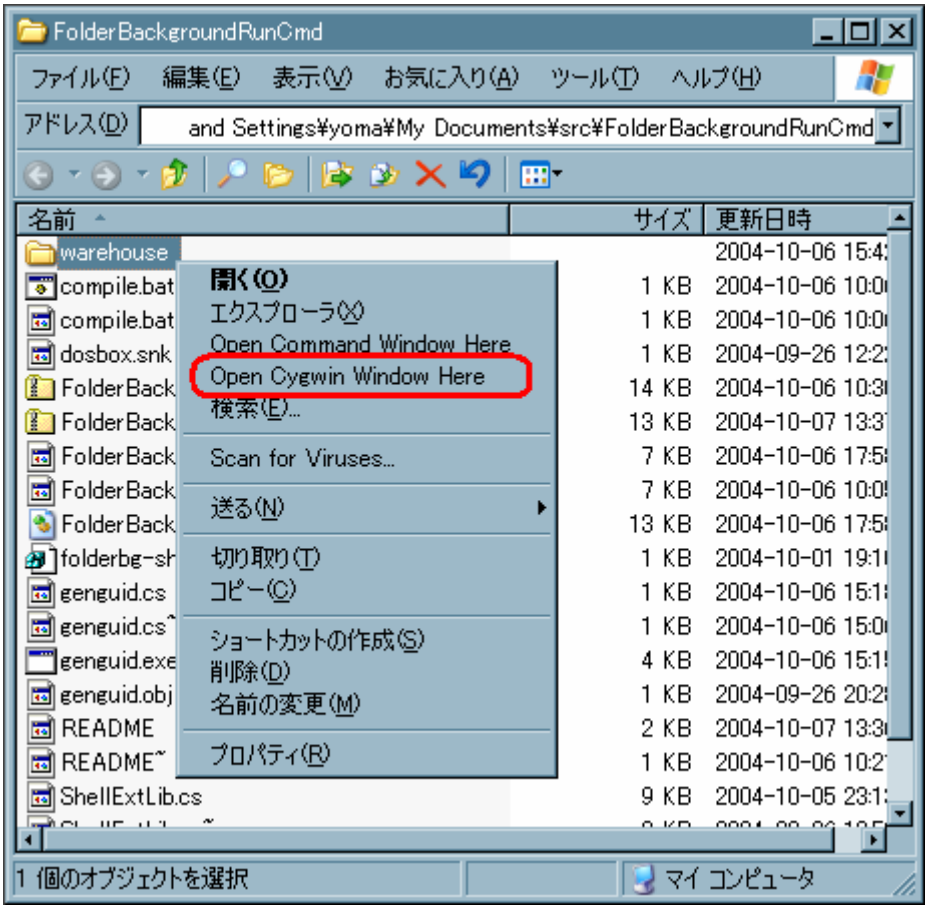
5 Open Cygwin Window Here の登録

それでは実際にレジストリを操作して、フォルダを右クリックした際のメニュー項目に Cygwin のシェルのウィンドウを開く項目を登録してみよう。言うまでもないことだが、レジストリの操作は適宜バックアップを取るなどして慎重に行っていただきたい。

まずメニュー項目は「Open Cygwin Window Here」としよう。そして、この項目を登録するキーの名前は「cygwin」としよう。すると以下の操作をすることになる。

1. regedit を起動して以下のキーを選択する。
   HKEY_CLASSES_ROOT¥Directory¥shell
2. 「編集→新規→キー」を選択し、キー名を「cygwin」と入力する。
3. cygwin キーの既定の値が regedit の画面の右側に表示されているので、「(既定)」をダブルクリックして「Open Cygwin Window Here」と入力する。
4. この時点ではまだ cygwin キーが開かれた状態なので、再び「編集→新規→キー」を選択し、キー名を「command」と入力する。「cygwin」というキー名は重要ではなかったのに比べて、「command」というキー名には意味があり、キー名が違ってはいけない。

まだ、「Open Cygwin Window Here」が選択された際に起動されるコマンド行の登録をしていない。しかし、ここまでの操作でフォルダを右クリックした際のメニューに「Open Cygwin Window Here」が表示されるようになった(図 5)。コマンド行が登録されていないので、「Open Cygwin Window Here」をメニューから選ぶと図 6のエラーが表示されるが。



[folder-cmenu-w-open-cygwin-here-anno.png]

図 5

[no-cmd-error.png]

図 6

6 Open Cygwin Window Here がどんなコマンド行を実行すべきか

Open Cygwin Window Here がメニューから選択された場合に実行されるコマンド行として、何を登録すべきだろうか。Open Command Window Here では「cmd.exe /k "cd %L"」が登録されている。CygTerm ではこのようにはいかない。なぜなら、cygterm.exe ではコマンド行引数でカレントフォルダを指定する方法がないからである。したがって Open Cygwin Window Here

から起動するプログラムが以下を行う必要がある。

1. カレントフォルダをコマンド行引数に従って設定する。
2. CygTerm を起動する。

このようなプログラムを「CygTerm 起動プログラム」と呼ぶことにしよう。

6.1 BAT ファイルを起動すると

CygTerm 起動プログラムは何で記述すべきだろうか。BAT ファイルで書いても一応目的は達せられるが実用的ではない。BAT ファイルを実行するとコマンドプロンプトウィンドウが現れ、それは Cygwin のシェルを終了するまで存在し続けるからである。具体的には以下のようになる。

以下の内容を C:¥cygwin¥open-cygwin-here.bat というファイルに保存したとしよう。

```
chdir %1
"C:¥Program Files¥cygterm¥cygterm.exe"
```

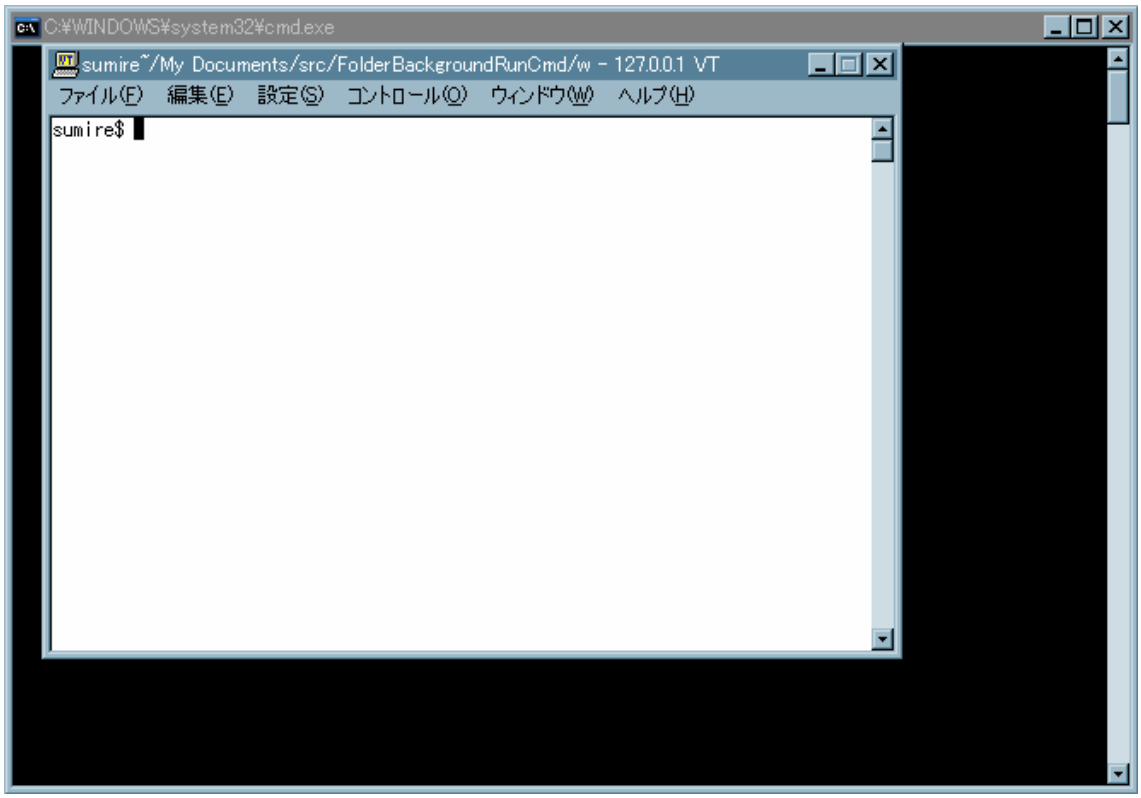
2 行目全体がダブルクォート(")で囲まれている理由は、パスがスペースを含んでいるからである。ダブルクォートで囲まないとエラーになる。そして、

HKEY_CLASSES_ROOT¥Directory¥shell¥cygwin¥command キー

の既定の値として以下を登録したとしよう。

```
C:¥cygwin¥open-cygwin-here.bat "%L"
```

こうした上で、フォルダを右クリックして「Open Cygwin Window here」を選ぶと図 7のようになるのである。CygTerm のウィンドウが存在する間、コマンドプロンプトウィンドウも存在し続ける。



[open-cygwin-here-bat.png]

図 7

6.2 コマンドプロンプトウィンドウを避けるには

コマンドプロンプトウィンドウが表示されないようにするには、コマンドプロンプトウィンドウを必要としないように CygTterm 起動プログラムを書けばよい。C++をはじめとして何で書いてもいいのだが、たとえば VBScript で書くと以下のようになる。

```
Option Explicit
Dim objArg
Set objArg = WScript.Arguments
Dim objSh
Set objSh = CreateObject("WScript.Shell")
objSh.CurrentDirectory = objArg(0)
objSh.Run """C:¥Program Files¥cygterm¥cygterm.exe"""
```

これを C:¥cygwin¥open-cygwin-here.vbs という名前で保存したとしよう。すると、HKEY_CLASSES_ROOT¥Directory¥shell¥cygwin¥command キーの既定の値として以下を登録することになる。

```
wscript.exe C:¥cygwin¥open-cygwin-here.vbs "%L"
```

ここで%L を「"」（ダブルクォート）で囲む理由は、そうしないとスペースを含むフォルダ名が正しく扱われるようにするためである。

6.3 /etc/profile の調整

cygterm.cfg で SHELL を以下のように設定している場合は更に作業が必要である。

```
SHELL = /bin/bash --login -i
```

/bin/bash を--login オプション付きで起動すると、/etc/profile を読み込んで実行する。/etc/profile の最後の行は以下の通りである。

```
cd "$HOME"
```

この行を削除する必要がある。

別の対処方法として、cygterm.cfg の SHELL =の部分から—login を削除することも考えられるが、/etc/profile での設定は有用なものが多いので、CygTerm から bash が起動される際、すなわち Cygwin 以外の世界から CygTerm が起動され、そこから更に bash が起動される場合は、--login があるほうが便利だと思う。

以上の作業でフォルダを右クリックして「Open Cygwin Window Here」をメニューから選ぶことで、指定したフォルダをカレントディレクトリとするシェルを起動することができるようになった。

7　ディスクドライブの右クリックでも

「マイ コンピュータ」を開くと、ハードディスクやリムーバブルドライブが表示される。CmdHere がインストールされていると、ドライブを右クリックした際に「Open Command Window Here」がメニューに表示される。これは以下のレジストリ・キーで制御されている。

HKEY_CLASSES_ROOT¥Drive¥shell¥cmd

したがって、ドライブについても「Open Cygwin Window Here」を行いたい場合は、以下のレジストリ・キーに登録を行う必要がある。

HKEY_CLASSES_ROOT¥Drive¥shell¥cygwin

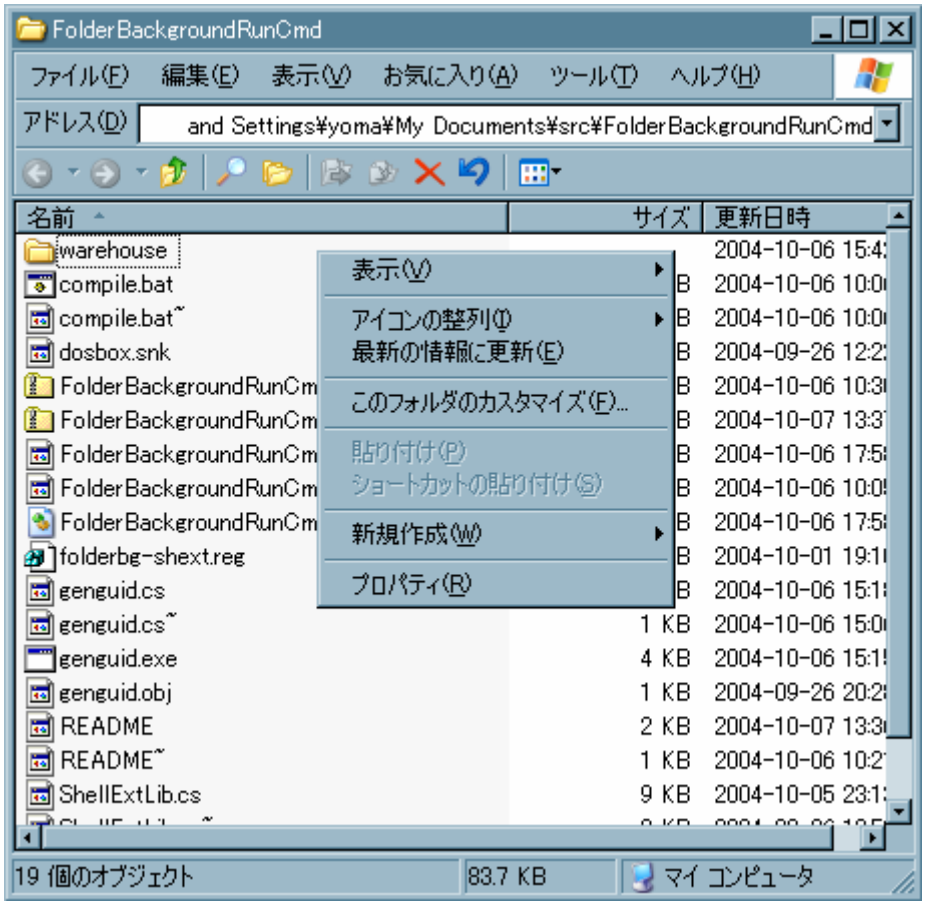
8　残る課題

さて、ここまでで「Open Command Window Here」と同等の「Open Cygwin Window Here」が実現できた。これで本当に十分だろうか。私にとっては十分ではなかった。

たとえば図 1のフォルダウィンドウが表示されている場合、このフォルダウィンドウのフォルダ「FolderBackgroundRunCmd」をカレントディレクトリとした Cygwin のシェルを起動したいことが多いからである。FolderBackgroundRunCmd フォルダの中にあるフォルダたとえば「warehouse」をカレントディレクトリとする Cygwin のシェルを起動したいことはあまりない。

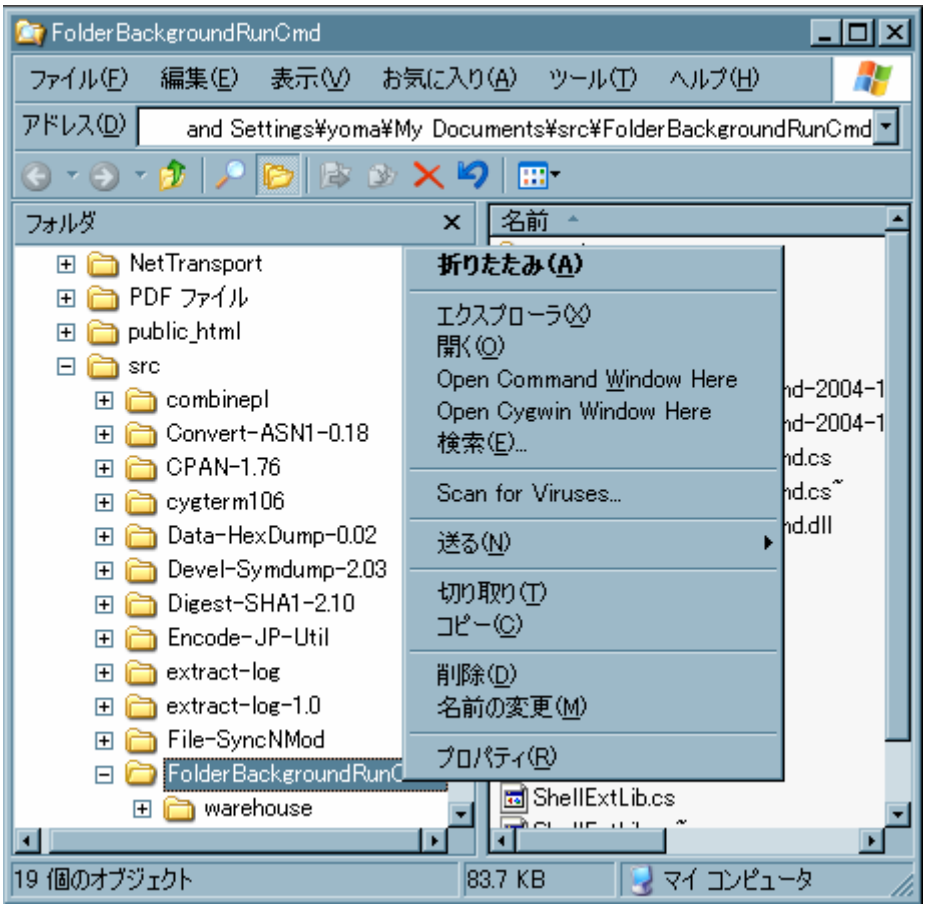
具体的な希望はこうだ。フォルダの内容表示の背景を右クリックした際のメニューに「Open Cygwin Open Here」があって、それを選択するとフォルダウィンドウのフォルダをカレントディレクトリとする Cygwin のシェルが起動して欲しいのである。しかし、実際にはフォルダの内容表示の背景を右クリックして表示されるメニューには「Open Command Window Here」も「Open Cygwin Window Here」もない（図 8）。

そこでたとえば図 9のようにフォルダの階層表示をさせて、そこで右クリックをすれば「Open Cygwin Window Here」がメニューに表示されるので、それで Cygwin のシェルを起動し、フォルダの階層表示を解除するという操作が考えられる。



[folderbg-cmenu.png]

図 8



[folder-pane-cmenu.png]

図 9

しかし、これではやはり不便である。フォルダの内容表示部分の背景で右クリックした際のメニューに「Open Cygwin Window Here」が現れて欲しい。

9 レジストリ操作で解決できるか

regedit の表示を眺めていると以下のキーがあることが分かった（図 10）。

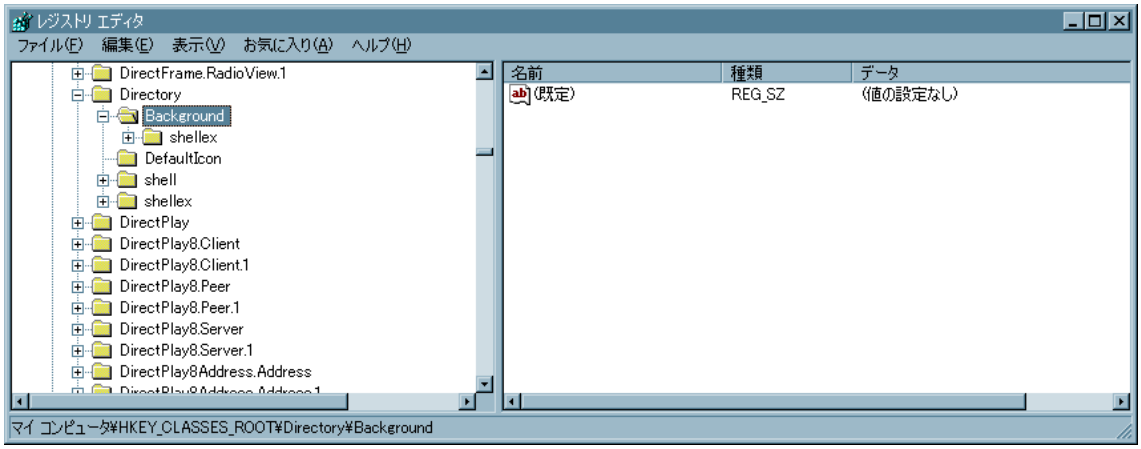
HKEY_CLASSES_ROOT¥Directory¥Background

このサブキーとして「shell」を登録し、その中に以下のキーの内容と同じものを登録すれば課題が解決できるのではないか。

HKEY_CLASSES_ROOT¥Directory¥shell

やってみたが、だめだった。図 8のままで変化が見られなかった。以下のキーは確かに使われているだけにとても残念だった。

HKEY_CLASSES_ROOT¥Directory¥Background¥shellex



[dirbg-reg.png]

図 10

10 シェル拡張

レジストリ登録だけでは実現できないとするとどうすればいいだろうか。

Windows Explorer

=

フォルダウィンドウを表示しているプログラム

=

Windows のシェル

であり、Windows Explorer には、Open Command Window Here のような機能拡張のほかに、もっと柔軟な機能拡張の手段があり、それはシェル拡張（shell extension）と呼ばれている。シェル拡張を自分で作成して登録すればよさそうだ。

しかし､本当にシェル拡張の枠組みを使ってフォルダ内容表示の背景での右クリックを拡張できるのであろうか。それを確かめようと「"background of a folder" context menu」を Google 検索したところ、「ShellToys」というシェアウェアが見つかった。説明にはフォルダ内容表示の背景での右クリックに対しても有効であると書いてある｡インストールしてフォルダ内容表示の背景で右クリックすると図 11のように確かにメニューが拡張されている。そして、以下のレジストリ・キーが新たに登録されていた。

HKEY_CLASSES_ROOT¥Directory¥Background¥shellex¥ContextMenuHandlers¥CFiExtensions

これにより、シェル拡張を作成し、その情報を以下のレジストリ・キーのサブキーに登録すれば目標が達成できることが確認できた。

HKEY_CLASSES_ROOT¥Directory¥Background¥shellex¥ContextMenuHandlers

[folderbg-cmenu-w-shelltoys.png]

図 11

11 作成したシェル拡張の仕様

いろいろ調べながら開発を進めた結果、課題を解決するシェル拡張を作成できた。その仕様を説明しよう。以下、そのシェル拡張を FBRC 拡張と呼ぶことにする。FBRC は「Folder Background Run Command」の意味である。

フォルダを右クリックした際のメニュー表示は図 5のように以下の 2 項目が追加されているとしよう。

Open Command Window Here

Open Cygwin Window Here

その状況で、FBRC 拡張をインストールすると、フォルダ内容表示の背景を右クリックした際のメニューが図 12のようになる。つまり、フォルダの右クリックのメニューに追加された項目がフォルダ内容表示の背景の右クリックのメニューにも追加されていて、かつ動作する。

より具体的に言えば、FBRC 拡張は以下のレジストリ・キーから追加メニュー項目の情報を得る。

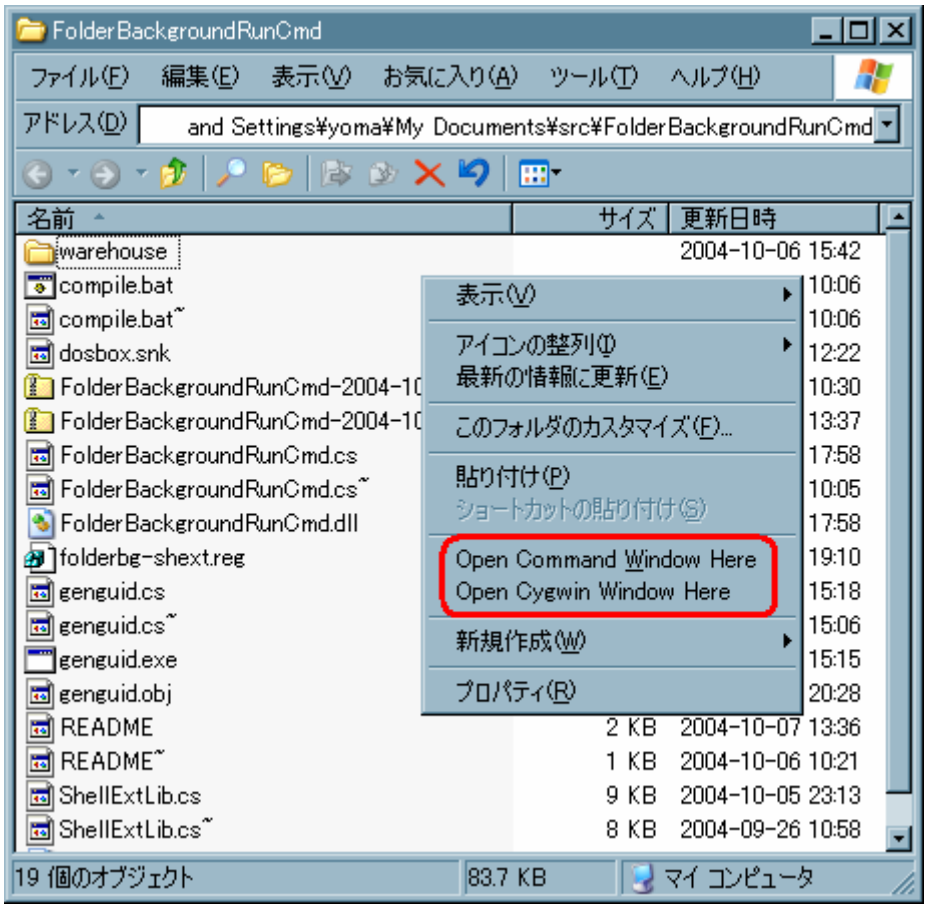
HKEY_CLASSES_ROOT¥Directory¥shell

このレジストリ・キーのサブキーのうち、既定の値が空のもの、たとえば find、は無視する。各サブキーの解釈はフォルダを右クリックした際のメニューと同様である。

更に図 13のように、フォルダウィンドウのファイルメニューにも項目が追加され、かつ動作する。また、デスクトップの何もないところで右クリックした際のメニューにも図 14のように項目が追加され、追加された項目を選択すると、自分のデスクトップ・フォルダをカレントディレクトリとする Cygwin のシェルなりコマンドプロンプトが立ち上がる。

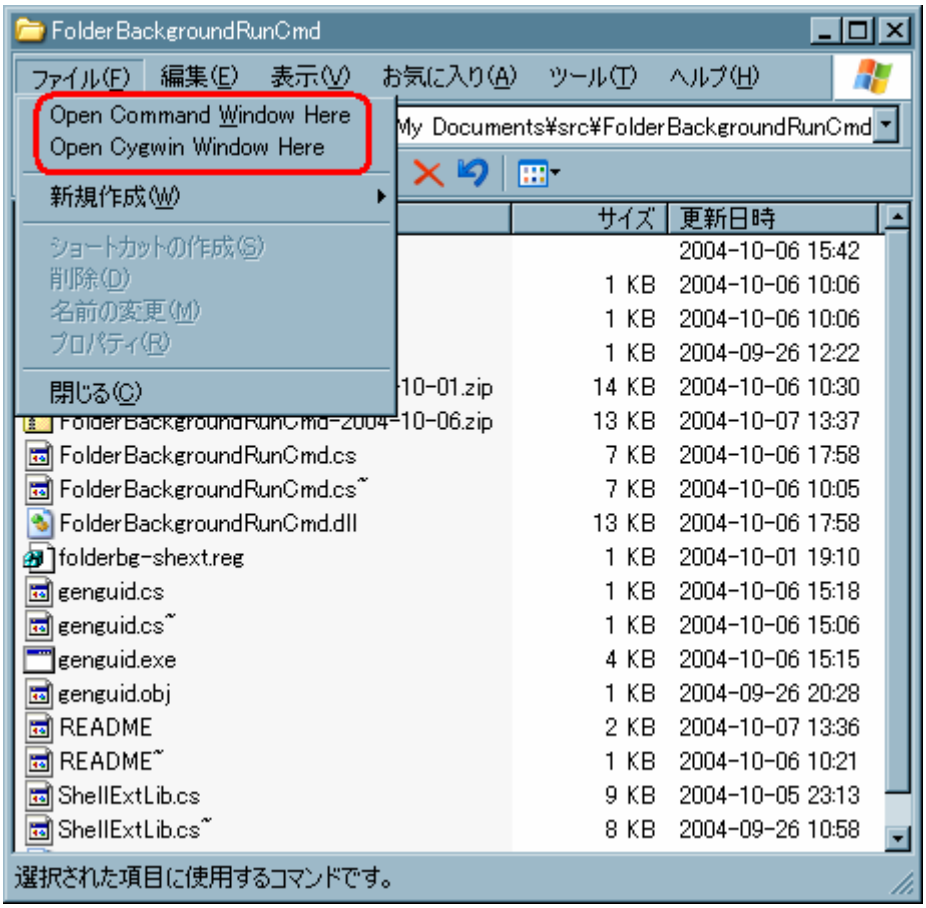
なお、FBRC 拡張がフォルダウィンドウのファイルメニューやデスクトップの右クリックでも有効なことは、特に意図したわけではない。以下のレジストリ・キーの内容がそれらのメニューについても作用を及ぼしているので自然にそうなったのである。

HKEY_CLASSES_ROOT¥Directory¥Background¥shellex¥ContextMenuHandler
s



[folderbg-cmenu-ext-anno.png]

図 12



[file-menu-ext-anno.png]

図 13

[desktop-cmenu-ext-anno.png]

図 14

12 インストール

12.1 準備

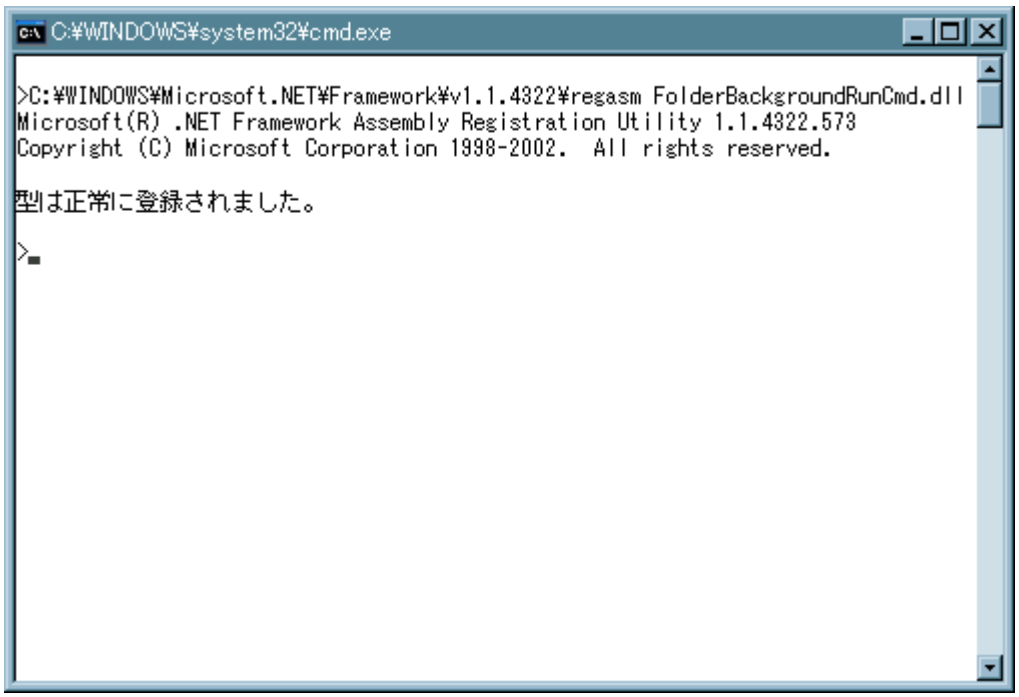
FBRC 拡張は C#で書いてあり、コンパイル済みのものをインストールするのであっても.NET Framework 1.1 が必要である。.NET Framework 1.1 がない場合はマイクロソフトのウェブサイトからダウンロードしてインストールしていただきたい。そして、PATH に以下のフォルダが含まれているとインストール作業が少し楽になる。

C:¥WINDOWS¥Microsoft.NET¥Framework¥v1.1.4322

12.2 実際のインストール

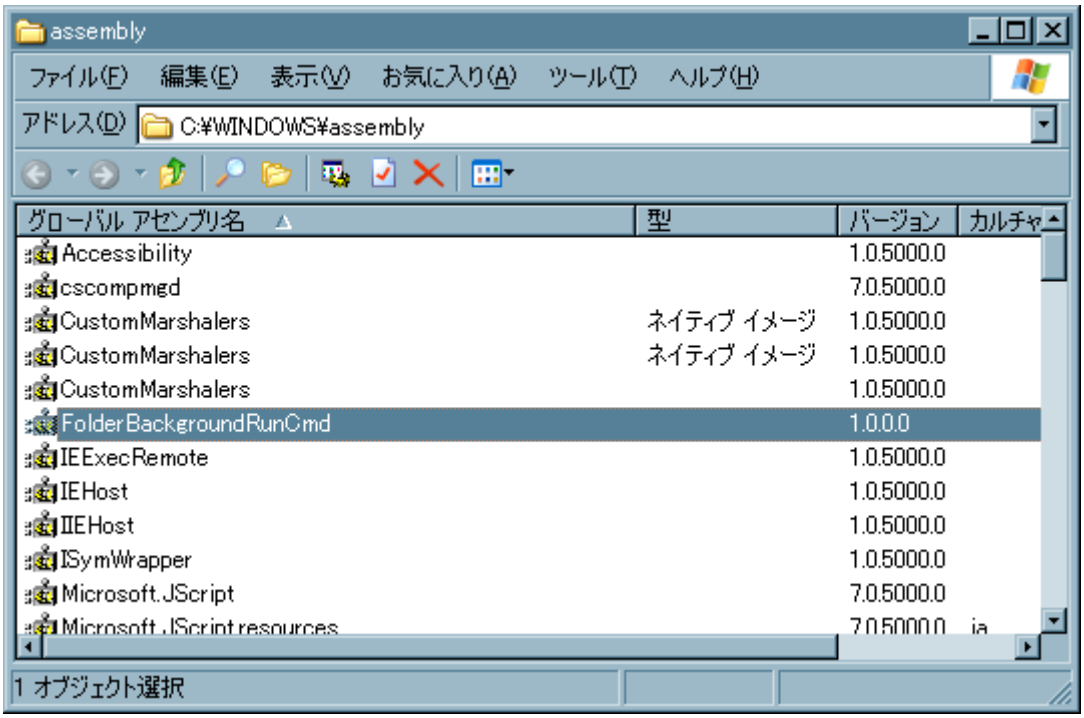
コンパイル済みの FBRC 拡張が、この記事の最初で紹介した URL から入手可能になっている。インストールの手順は以下の通りである。

1. FolderBackgroundRunCmd-pkg*.zip ダウンロードし、展開する。
2. コマンドプロンプトウィンドウを開き展開したファイルが置いてあるフォルダに移動する。
3. 以下のコマンド行を実行する(図 15)。
   C:¥WINDOWS¥Microsoft.NET¥Framework¥v1.1.4322¥regasm FolderBackgroundRunCmd.dll
   ただし、PATH に C:¥WINDOWS¥Microsoft.NET¥Framework¥v1.1.4322 が含まれていれば、regasm をフルパスで指定する必要はない。
4. シェル拡張の登録に必要なレジストリ操作が上記の操作で行われるはずだが、.NET Framework の設定によってはレジストリ操作に失敗する場合がある。その場合は、展開したファイルに含まれている folderbg-shext.reg をダブルクリックすることでレジストリ操作を行う。
5. FolderBackgroundRunCmd.dll を Global Assembly Cache に登録する。具体的には C:¥WINDOWS¥Assembly のフォルダウィンドウを開いて、そこに FolderBackgroundRunCmd.dll をドラッグアンドドロップする(図 16)。
6. これで FBRC 拡張が動作するようになったはずだ。そうでない場合は Windows からログオフしてログオンする。



[regasm-cmdprompt.png]

図 15



[gac-w-folderbgruncmd.png]

図 16

12.3 アンインストール

FBRC 拡張をアンインストールするには、以下を行う。

1. ダウンロードしたファイルを展開したフォルダで以下のコマンド行を実行する。
   regasm /unregister FolderBackgroundRunCmd.dll
2. C:¥WINDOWS¥assembly から FolderBackgroundRunCmd を削除する。

13 動作のしくみ

次に FBRC 拡張が動作するしくみを説明しておこう。

FBRC 拡張の実体は FolderBackgroundRunCmd.dll である。このファイルは C#のプログラムをコンパイルして作ったものである。インストール作業の結果以下のレジストリ・キーが作成されている。

HKEY_CLASSES_ROOT¥Directory¥Background¥shellex¥ContextMenuHandlers¥RunCmd

そして、その既定の値が以下ようになっている(図 17)。

{2DA8923C-8A40-4D01-95C9-F208491ACBC6}

「2DA8923C-8A40-4D01-95C9-F208491ACBC6」は FBRC 拡張の CLSID(class identifier)である。CLSID は Windows 上の COM(Component Object Model)オブジェクトのクラスを識別する。

見ての通り CLSID は 16 バイトの値で、新たに作られた COM オブジェクトのクラスに対する CLSID は、CoCreateGuid()という Win32 API の関数により作成することになっている。CoCreateGuid()は呼び出すたびにランダムな値を返す。CLSID の空間は 16 バイト=128 ビットとかなり広いので、一元管理しなくても CLSID の衝突が起こる可能性は非常に低いということなのだろう。

フォルダ内容表示の背景で右クリックしたときや、フォルダウィンドウでファイルメニューを呼び出した際に、Windows Explorer(=フォルダウィンドウを表示しているプログラム)は、以下のレジストリ・キーのサブキーを順に調べて、登録されている CLSID の COM オブジェクトを順に呼び出す。

HKEY_CLASSES_ROOT¥Directory¥Background¥shellex¥ContextMenuHandlers

つまりシェル拡張は COM オブジェクトのクラスであり、レジストリにその CLSID を登録することにより、Windows Explorer から呼び出されるようになるのである。

CLSID を基に COM オブジェクトが呼び出されるしくみは次のとおりである。

まず以下のレジストリ・キーが参照される。

HKEY_LOCAL_MACHINE¥SOFTWARE¥Classes¥CLSID¥{*CLSID*}

たとえば FBRC 拡張については以下のレジストリ・キーである(図 18)。

HKEY_LOCAL_MACHINE¥SOFTWARE¥Classes¥CLSID¥{2DA8923C-8A40-4D

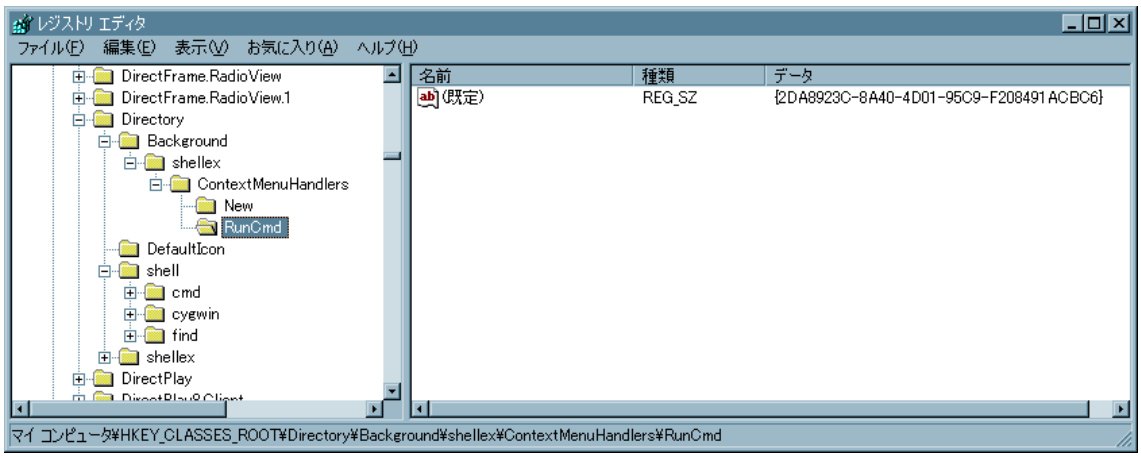
01-95C9-F208491ACBC6}

そのレジストリ・キーの中の InprocServer32 サブキーの既定の値（FBRC 拡張については「mscoree.dll」）から、どの DLL ファイルでその COM オブジェクトのクラスが実装されているかが分かり、その DLL ファイル中のコードが呼び出される。

.NET Framework を使わずに実装されている COM オブジェクトのクラスについては、ここまでなのだが、.NET Framework で実装されているクラスについては更に先がある。COM オブジェクトを呼び出す側から見ると、その COM オブジェクトのクラスが.NET Framework を使っていようがいまいが呼び出し方は変わらないのだが。

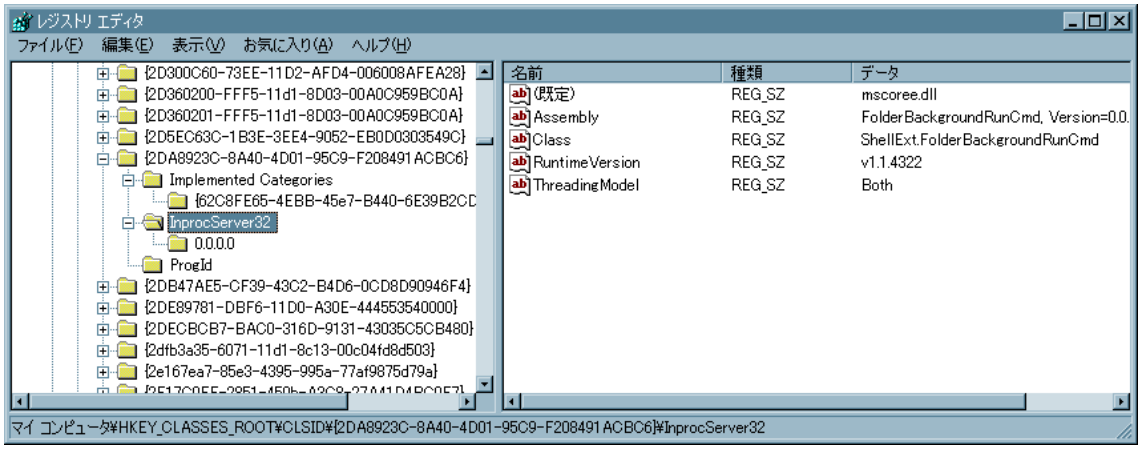
mscoree.dll は.NET Framework の CLR(Common Language Runtime)の実体である。.NET Framework で実装された COM オブジェクトのクラスはすべてエントリ・ポイントが mscoree.dll(=CLR)になっていて、ある COM オブジェクトの処理のために呼び出された CLR は、その COM オブジェクトのクラスがどの.NET アセンブリ中のどのクラスで実装されているかを判断して、対応するコードを実行する。図 18に、FBRC 拡張のクラスが.NET アセンブリとその中のクラスに対応付けされている様子が示されている。

そして CLR は Global Assembly Cache に登録されている FolderBackgroundRunCmd.dll の中の、ShellExt.FolderBackgroundRunCmd クラスを呼び出す。



[folderbg-shex-runcmd-reg.png]

図 17



[folderbgruncmd-shext-clsid-reg.png]

図 18

14 C#を使うに至った経緯

ここで、FBRC 拡張を C#で書くに至った経緯を説明しよう。

先に述べたようにフォルダ内容表示の背景の右クリックのメニューの拡張がレジストリ登録ではできない。目的を達するためにはシェル拡張を作る必要がある。シェル拡張とは COM オブジェクトのクラスである。COM オブジェクトのクラスを実装する言語として私がまず考えたのは C++だった。Visual C++ 2003 Toolkit が Microsoft から無料で配布されており、C++のコンパイルには何の問題もない。そこで、C++で書かれたシェル拡張の例を探してみた。いくつか見つかったが、いずれも製品版 Visual C++を使うことを前提としており、ATL(Active Template Library)がないとコンパイルできない。残念ながら ATL は無償配布されていない。やろうとしているような環境改善のためにそれほど安くないツールを購入するのはためらわれる。MinGW で COM オブジェクトのクラスを実装している例はないかと思って少し探したがそれも見つからなかった。

C#で書かれたシェル拡張の例を探したところ、いくつか見つかり、中でも「Manage with the Windows Shell: Write Shell Extension with C#」と題するものが大いに参考になった。その URL は以下のとおりである。

http://www.theserverside.net/articles/showarticle.tss?id=ShellExtensions

そのページで説明されているシェル拡張のソースコードをダウンロードして.NET Framework 付属の C#コンパイラ[2]でコンパイルし、そのページの指示に従ってインストールしたところ動作した。ソースコードを若干変更する必要はあったが。

これで、私の環境で、C#でシェル拡張を実装して利用することができることが分かった。私が FBRC 拡張を C#で実装できたのは、その記事によるところが大きい。

---

[2] C:¥WINDOWS¥Microsoft.NET¥Framework¥v1.1.4322¥csc.exe

15 プログラムの概要

この記事では FBRC 拡張のソースコードについて詳しい説明はしない。しかし、C#にそれほど詳しくなくても以下の説明を読み、また、関連する情報を Web ページで読めば FBRC 拡張を改造することはできるだろう。また、C#になじみのない人に C#のプログラムでどんな処理が、どんな記述で可能かの雰囲気を知ってもらえればと思っている。

FBRC 拡張は、この記事の最後に示すリスト 1(FolderBackgroundRunCmd.cs)とリスト 2(ShellExtLib.cs)で実装されている。シェル拡張の機能の記述はもっぱらリスト 1 で行っており、リスト 2 は主に Win32 API の関数を C#から呼び出すために必要な定数や構造体の定義を行っている。このように、C#からは Win32 API の関数がかなり自由に呼び出せるのである。

以下、リスト 1 中の各メソッドの働きを簡単に説明しておく。名前が「IShellExtInit.」あるいは「IContextMenu.」で始まっているものは FBRC 拡張のようなシェル拡張、つまりコンテキストメニューハンドラでは、必ず実装しなければならない。

IShellExtInit.Initialize()

右クリックによって、FBRC 拡張のオブジェクトのインスタンスが作られ、そのインスタンスに対して最初に適用されるメソッドである。この中でメニューに追加する項目をレジストリから読み込んでいる。このことから、レジストリにメニュー項目を追加すれば、それがすぐに反映されることになる。

IContextMenu.QueryContextMenu()

右クリックによって表示されるメニューを準備している段階で呼び出されるメソッドである。メニューに追加する項目のデータを設定している。また、メニュー項目の境界線も設定している。

IContextMenu.GetCommandString()

追加したメニュー項目上にマウスカーソルがあるときにフォルダウィンドウのステータスバー(内容表示の下の部分)に表示する内容などを返す。

ExecuteCommand()

IContextMenu.InvokeCommand()からのみ呼び出されるメソッドで、名前の通り、コマンドを実行する。

IContextMenu.InvokeCommand()

追加したメニュー項目が選択された際に呼び出されるメソッドである。

RegisterServer()

regasm が.NET アセンブリを登録する際に呼び出すメソッドである。登録するアセンブリが動作するのに必要な処理を記述しておけば regasm が自動的に実行してくれる。

UnregisterServer()

regasm /unregister で.NET アセンブリの登録を抹消する際に呼び出されるメソッドである。

16 コンパイル方法

FBRC 拡張を C#のソースからコンパイルするには以下のコマンド行を実行する。

```
csc /t:library FolderBackgroundRunCmd.cs ShellExtLib.cs
```

ただし、このためには、fbrc.snk というファイルが FolderBackgroundRunCmd.cs と同じフォルダになければならない。fbrc.snk はソースコードの配布に含まれている。

自分で fbrc.snk を作ってみたいという人は、無料で配布されている.NET Framework SDK に含まれる sn.exe というコマンドを以下のコマンド行で実行すればよい。

```
sn -k fbrc.snk
```

17 おまけ

たとえばこの記事を参考に新たなシェル拡張を作成するとしよう。すると、そのシェル拡張の CLSID が必要になる。リスト 3 に、CLSID を生成して標準出力に出力してくれるプログラムを示す。コンパイルは「csc genguid.cs」とすればよい。genguid.exe にコマンド行引数はない。

ここで CLSID についてもう少し説明しておこう。Windows では COM オブジェクトのクラス以外にも様々なものを 16 バイトの値で識別している。その 16 バイトの値を GUID(Globally Unique Identifier)と呼ぶ。CLSID は GUID の 1 つの利用形態なのである。というわけで、リスト 3 のプログラムは genclsid ではなく、genguid という名前なのである。

☆

Cygwin の使い勝手の向上のために、かなり Windows に踏み込んだことをすることになってしまった。UNIX をずっと使ってきたが、最近は Windows も使わなければならない私のような人にとっては、この記事は Windows でちょっと突っ込んだことをするためのよいきっかけあるいは手がかりになるのではないか。そうなれば喜ばしい限りである。

（よしだ・まさひで）