4D System Tool

By Jeremy Sullivan, Developer, 4D Inc.

TN 06-11

Introduction

このテクニカルノートでは、LAUNCH EXTERNAL PROCESS の使用方法を説明しています。4th Dimension 2004 で追加された LAUNCH EXTERNAL PROCESS コマンドは、極めて高いポテンシャルを有する新しいコマンドです。

Mac OS X の根底にある UNIX レイヤには、LAUNCH EXTERNAL PROCESS で利用できる多数のアプリケーションやユーティリティが存在します。基本的にいって、ターミナルで実行できることは、すべて LAUNCH EXTERNAL PROCESS で実行することができます。

Windows の場合、LAUNCH EXTERNAL PROCESS ですべての DOS コマンドを実行することができます。

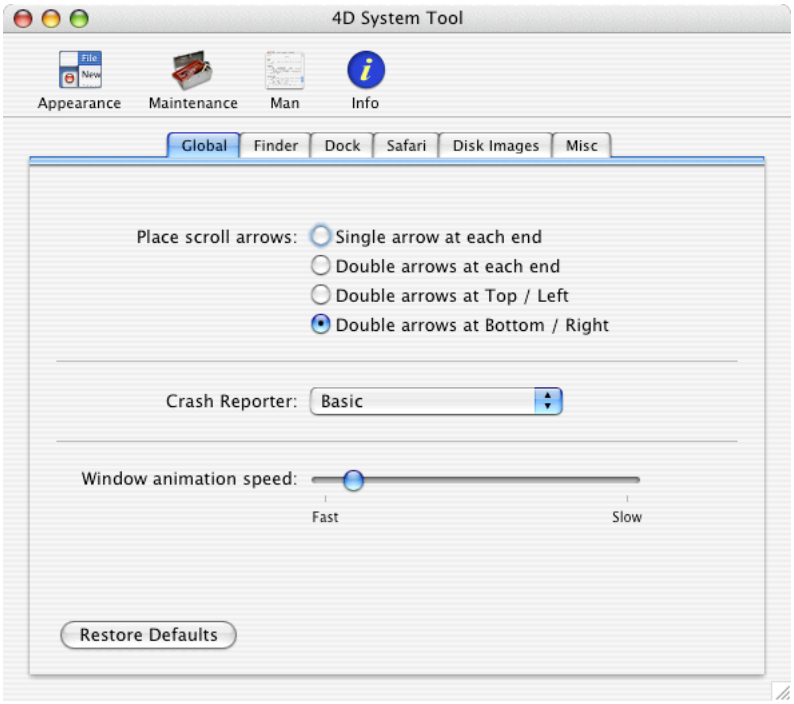
About the Example

付属のサンプル 4D System Tool アプリケーションは、ふたつの部分に分かれています。今回はシステムコールが関係しているため、クロスプラットフォーム要素はあまりなく、プラットフォームによって異なるインタフェースおよび実行コードが選択されるようになっています。サンプルをすべて試したいのであれば、Mac OS X と Windows XP の両方で実行してください。

サンプルは、Mac OS X 10.2 および Windows XP で実行されることが想定されています。各 OS の他のバージョンについては、テストを実施していません。

Using the Example – Mac OS X

サンプルは、Mac OS X で実行した場合、Cocktail、Onyx、TinkerTool のようなアプリケーションとして動作します。画面の上部にあるツールバーボタンによって Appearance、Maintenance、Man、Info という 4 種類のツールを利用することができます。



Appearance ツールでは、Finder、Dock、Safari などのアプリケーションの設定を変更することができます。たとえば、スクロールバーの表示形式、Dock をしまうときのエフェクト、アイコンサイズ、ディスクイメージの扱いなど、いろいろな設定を変更することができます。

Maintenance ツールでは、定期スクリプトの実行、システムソフトウェアのパーミッション復元、アプリケーションインストール後に実行される最適化処理であるプリバインディングの再試行、ブートディスク情報の確認などができます。

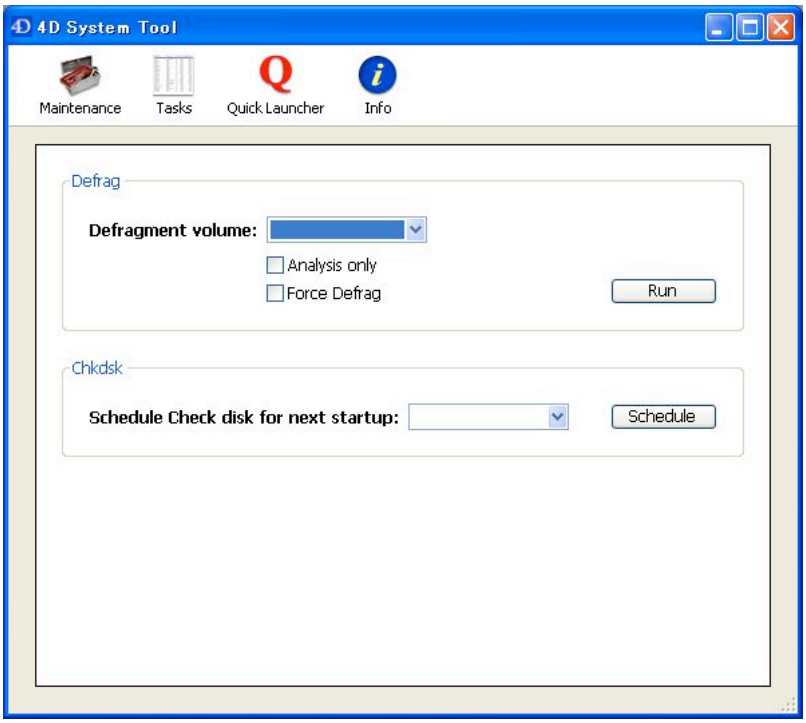
Man ツールでは、Man ページを表示し、PDF または HTML 形式で書き出すことができます。Man(Manual)ページには、すべての UNIX レイヤコマンドのドキュメントが収められています。

Info ツールでは、カーネルのバージョン、インストールされたメモリ容量、プロセッサの処理速度など、実行中のシステムに関するさまざまな情報を確認することができます。

さらに、ファイルメニューから Execute Command を選択すると、ダイアログが表示され、直接コマンドを入力して実行することができます。このダイアログでは、コマンドの実行によって意図せずにシステムを破損する可能性があるので、くれぐれも細心の注意を払ってください。

Using the Example – Win XP

サンプルを Windows で実行した場合、Mac OS X の場合とは別のコマンドが実行できます。画面の上部にあるツールバーボタンによって、Maintenance、Tasks、Quick Launcher、InfoInfo という 4 種類のツールを利用することができます。



Maintenance ツールでは、デフラグの実行、chkdsk または次回のスタートアップでチェックを実行するボリュームの設定ができます。

Tasks ツールでは、実行中プロセスのリストを表示し、任意のプロセスを終了することができます。要するに、Control+Alt+Delete を入力してタスクマネージャを表示するのと同じことができます。

Quick Launcher ツールを選択すると、Run コマンドで実行することのできるいろいろなツールのリストが表示されます。アプリケーションの追加/削除など、一部のツールは標準の XP インタフェースからも選択できますが、通常はコマンドプロンプトを使用しなければ実行できないものもあります。、Add または Remove ボタンを使用することにより、アプリケーションをリストに追加あるいは削除することができます。このリストは、Extras フォルダの中にある XML 形式の外部ファイルで管理されています。

Info ツールでは、Windows バージョン、インストールされたメモリ容量、プロセッサの処理速度など、実行中のシステムに関するさまざまな情報を確認することができます。

Getting Into the Code

サンプルでは、いずれの場合もプロジェクトメソッド ST_comm_executeCommand を実行することによってシステムアプリケーションをコールしています：

```
C_TEXT($1;$0;$command_t;$inputStream_t;$outputStream_t;$errorStream_t)
C_BOOLEAN($2;$needsAuthorization_b)
$command_t:=$1
$b_needsAuthorization:=$2
$inputStream_t:=""
$outputStream_t:=""
$errorStream_t:=""
If ($needsAuthorization_b)
   If (<>ST_userName_t="") | (<>ST_password_t="")
      ST_comm_authorize ` 認証情報を取得
      $inputStream_t:=<>ST_password_t
   Else
   $inputStream_t:=<>ST_password_t
   End if
End if
If (<>ST_ONWINDOWS_b)
   SET ENVIRONMENT VARIABLE("_4D_OPTION_HIDE_CONSOLE";"true")
End if
LAUNCH EXTERNAL PROCESS($command_t;$inputStream_t;$outputStream_t;$errorStream_t)
ST_error_setError ($errorStream_t)
```

$0:=$outputStream_t

メソッドは、ふたつのパラメータ、つまり実行するコマンドラインと認証の有無を示すブール値(OS X コマンドのみ)を受け取ります。

認証が必要であれば、まず以前にユーザから受け取ったユーザ名とパスワードが調べられます。ユーザ名とパスワードが保存されていない場合、入力を促すダイアログが表示されます。

次に、Windows であれば、コンソールウインドウを表示しないように SET ENVIRONMENT VARIABLE を使用して環境変数を設定します。

最後に、LAUNCH EXTERNAL PROCESS コマンドを実行し、エラーを受け取った場合はそのエラーを保存し、出力ストリームを呼出元メソッドに返します。

Code Examples

ST_comm_executeCommand メソッドの簡単な使用例は、ST_app_startup の 32 行です：

```
<>T_userName_t:=ST_comm_executeCommand ("whoami";False)
```

whoami とは、カレントユーザ名を返す簡単な UNIX ユーティリティです。スタートアップでこのコマンドをコールしているので、後にダイアログを表示したときには、カレントユーザ名があらかじめ入力されているようになっています。

複雑なコマンドを実行する場合、事前に必要なコマンドラインを構築する必要があります。良い例が ST_man_saveAsHTML メソッドの 51-54 行です：

```
$manPath_t:=ST_comm_executeCommand ("/bin/sh -c ¥"man -w"+$manPage_t+"¥"";False)
$manPath_t:=Substring($manPath_t;1;Length($manPath_t)-1)
ST_comm_executeCommand ("/bin/sh -c ¥"groff -Thtml -man "+$manPath_t+" >"+$documentPath_t+"¥"";False)
ST_comm_executeCommand ("/bin/sh -c ¥"open '"+$documentPath_t+"'¥"";False)
```

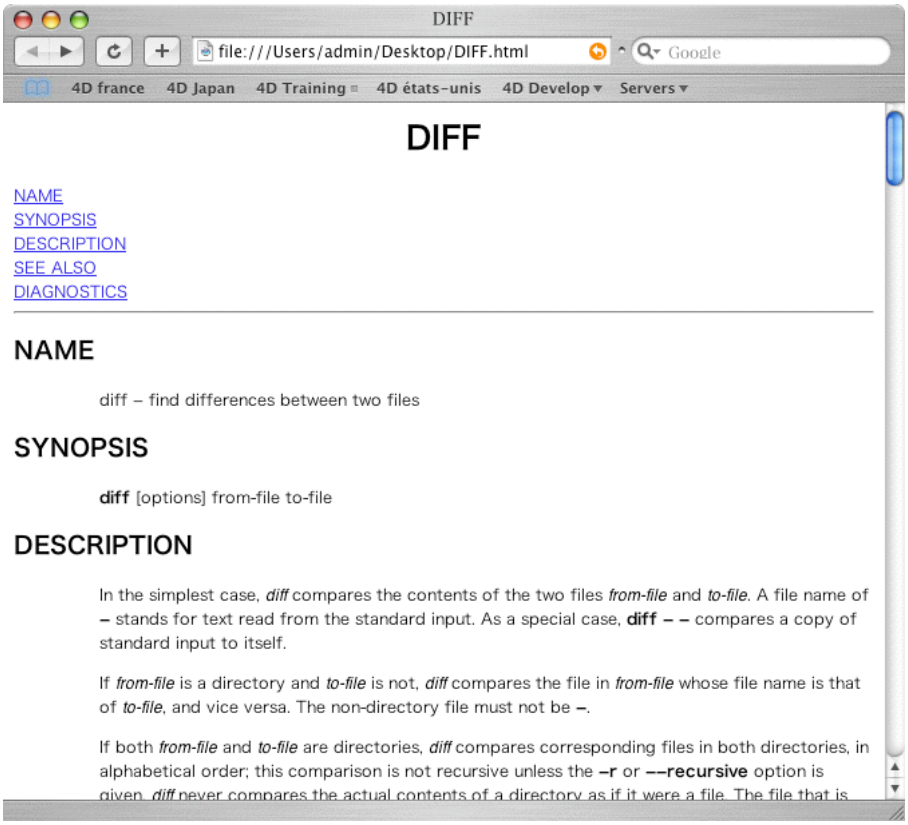
このように-c オプションをつけて/bin/sh を実行し、その際にコマンド名およびオプションを引用符でかこって渡すのがポイントです。このようにすれば、バッシュシェルに引用符でかこわれた文字列を実行させることができ、直接 LAUNCH EXTERNAL PROCESS に複雑なコマンドを渡した場合に生じる問題を回避することができます。

上記のメソッドの場合、groff をコールしている箇所以外に問題を起こしそうな部分はありませんが、そうであってもバッシュシェルの中でコマンドを実行することに損はありません。

コマンドの最初の行では、-w オプションをつけて man コマンドをコールしています。この場合、man コマンドはマニュアルページとして表示するファイルのパスを返します。

コマンドの次の行では、コールの際に追加されたラインフィードを取り払っています。
コマンドの次の行では、groff ユーティリティをコールしています。-Thtml オプションは HTML 形式で結果を返すこと、-man オプションはそれがマニュアルページであることを groff に対して指示しています。次のパラメータは、ファイルパスなので、事前に man コマンドで取得しておいたファイルパスを渡します。>記号は結果をファイルに書き出すことを意味します。最後のパラメータは書き出し先のファイルパスです。

コマンドの次の行では、出力ファイルを HTML タイプに関連付けられたアプリケーション、通常はデフォルトの Web ブラウザで開くためのものです。



Summary

サンプルをみれば、プラットフォーム特有のユーティリティを 4D に組み込む方法がなんとなく分かると思います。使用したのは利用できるコマンドの中のごく一部であり、とりわけ Mac OS X の場合、他にも便利なコマンドが豊富に存在します。

LAUNCH EXTERNAL PROCESS を使用すれば、実行中の環境に関する詳細な情報が取得できるという点も実証されました。

Mac OS X の場合、groff や curl(テクニカルノート 05-18 を参照してください)を活用したり、AppleScript や osascript を実行することもできます。

Windows の場合、すべての DOS コマンドが実行でき、相手側が対応していれば、プログラムにフラグを渡したりすることができます。

Mac OS X で利用することのできる UNIX コマンドの全容については、サンプルデータベースの Man ツールや、ManOpen などのアプリケーションを使用してマニュアルドキュメントを参照してください。(http://www.clindberg.org/projects/ManOpen.html)

DOS コマンドの完全なリストについては、Microsoft 社の Web サイトを参照してください。(http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ntcmds.mspx?mfr=true)