4D System Tool

By Jeremy Sullivan, Developer, 4D Inc.
TN 06-11

Introduction

This Tech Note shows several different examples using Launch External Process. It should give you an idea of the various possibilities this command makes available. Launch External Process was added to the language in 4D 2004.

Mac OS X's underlying Unix layer has many applications and utilities that you can access via Launch External Process. Almost anything you can do in the Terminal you can also accomplish with Launch External Process.

When running on Windows you have access to the complete set of DOS commands.

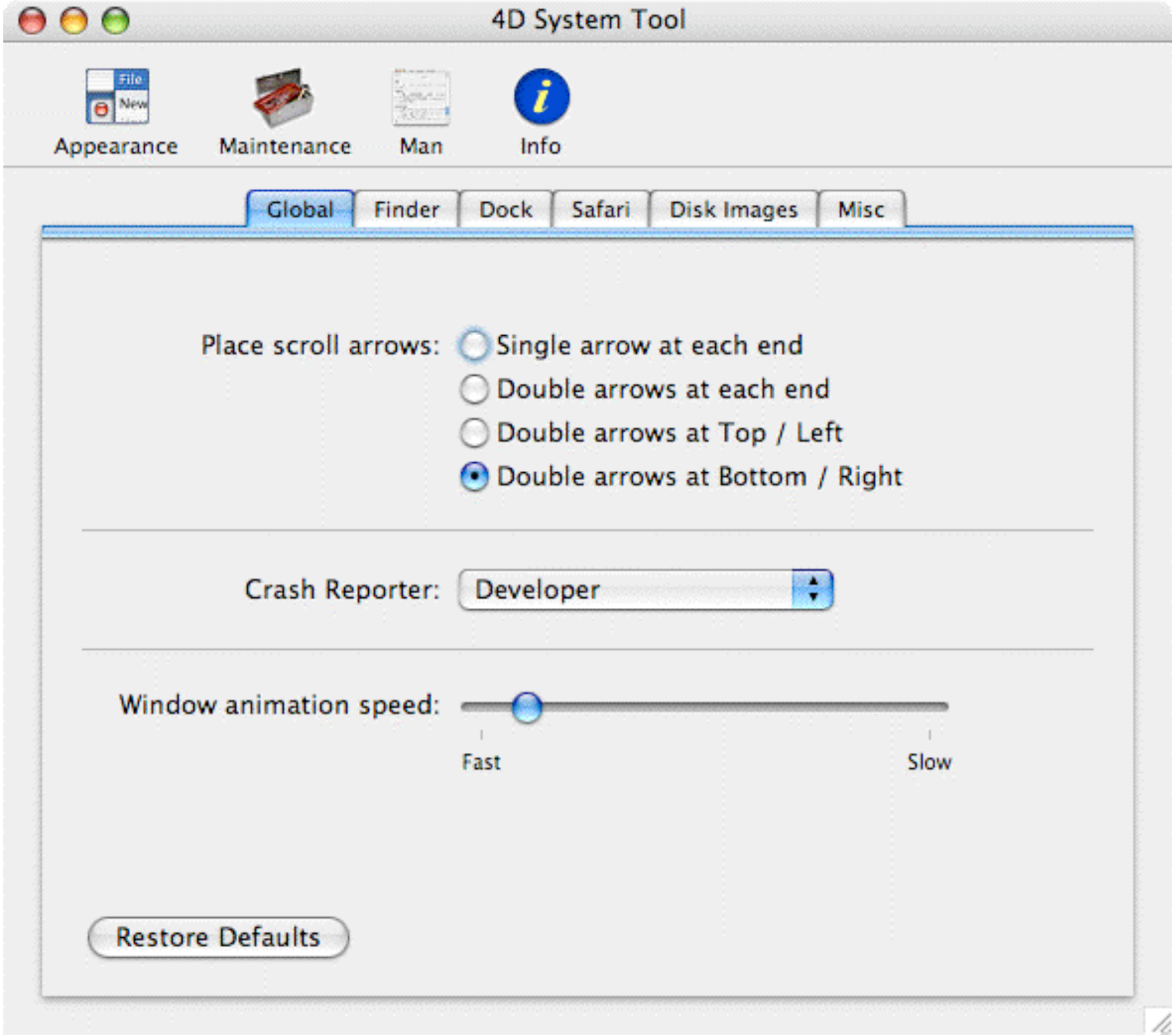
About the Example

4D System Tool is an application that is basically divided into two parts. Since we are dealing with system calls directly, there is very little that can be used cross platform. The interface and available commands are therefore completely different under each platform. To get the most out of the example you should run it on both Mac OS X and Windows XP if at all possible.

Note that this example assumes you are using Mac OS X 10.4 and/or Windows XP. None of the examples have been tested on prior versions of the respective operating systems.

Using the Example – Mac OS X

On Mac OS X the application is similar to utilities such as Cocktail, Onyx, or TinkerTool. When the application is launched you will have four tool bar buttons available to you: Appearance, Maintenance, Man, and Info.



Place scroll arrows:
○ Single arrow at each end
○ Double arrows at each end
○ Double arrows at Top / Left
● Double arrows at Bottom / Right

Crash Reporter: Developer

The Appearance tool allows you to change various settings in the Finder, Dock, Safari, and other applications. You can change the way scroll bars appear, the way the Dock behaves and displays its icons, disk image handling, and various other settings.

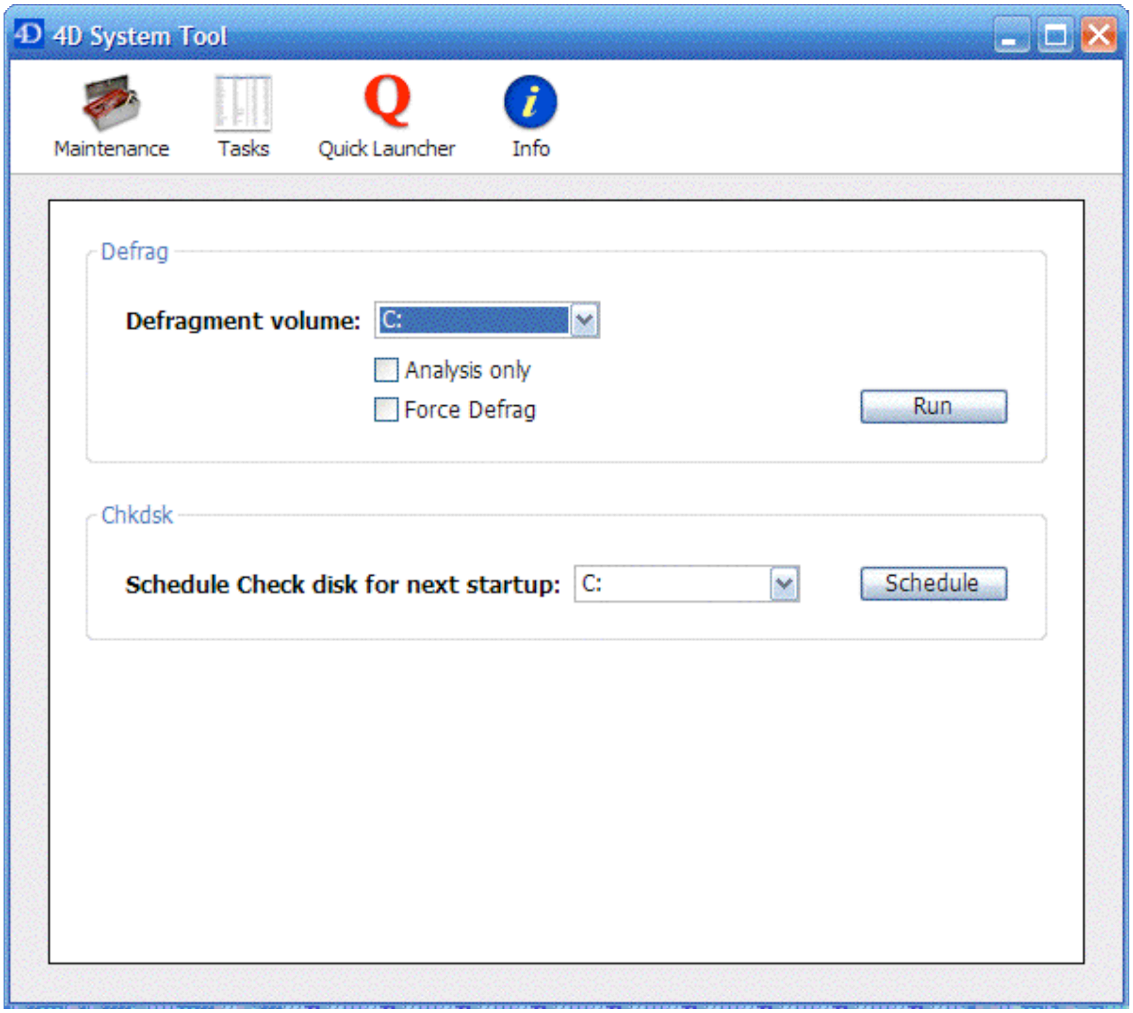
The Maintenance tool allows you to run the Periodic command, repair disk permissions, update pre-binding, and see some information about the boot disk.

The Man tool allows you to view the Man pages and save them as either PDF or HTML documents. Man (manual) pages are the documentation that is provided for all of the various commands available at the Unix layer.
The Info tool shows you various information about the system you are running on, such as the Kernel version, installed memory, processor speed, etc.

You can also select Execute Command from the File menu to open a dialog that will allow you to execute any command directly. Please be careful when using this tool as it will not prevent you from damaging your system if you execute the wrong command.

Using the example – Win XP

When the application is run on Windows XP you will have a different set of available commands. The four available tool bar buttons are: Maintenance, Tasks, Quick Launcher, and Info.



Defragment volume: C:

☐ Analysis only

☐ Force Defrag

Schedule Check disk for next startup: C:

With the Maintenance tool you can de-fragment a volume or set a volume to be checked by chkdsk on the next startup.

The Tasks tool will show you all of the currently running processes and allow you to quit a process. This is basically the same as the Task Manager when you type control-alt-delete.

The Quick Launcher displays a list of various tools that you can run from the Run command. Some of these are available to you through the standard XP interface, such as Add/Remove Programs, and others are normally only available by running them from the prompt. You can add or remove programs from the list using the Add and Remove buttons at the bottom of the window. The list of applications is saved in an xml file in the Extras folder.

The Info tool shows you various information about the system you are running on, such as Windows version, installed memory, processor speed, etc.

Getting Into the Code

Every call that is made to a system application goes through the project method ST_comm_executeCommand:

```
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 ` get authorization info
    $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
```

This method is passed two parameters: the command to execute and a boolean indicating if this command will need authorization (for OS X commands only).

If authorization is needed a check is done to see if the username and password have been previously supplied by the user. If not, a dialog opens that requests it.

When the application is running on Windows a call is made to SET ENVIRONMENT VARIABLE so the console won't open while the command is executing.

Finally the LAUNCH EXTERNAL PROCESS call is made, saving any error that is received and returning the result to the calling method.

Code Examples

A simple example of how to use ST_comm_executeCommand is in the project method ST_app_startup (line 32):

```
◊ST_userName_t:=ST_comm_executeCommand ("whoami";False)
```

whoami is a Unix utility that simply returns the current user's name. This is run at startup as a convenience to the user so they don't have to enter their user name when the authentication dialog is displayed.

To run more complex commands they need to be built so they will be properly executed. For an example look at the project method ST_man_saveAsHTML (lines 51-54):

```
$manPath_t:=ST_comm_executeCommand ("/bin/sh -c \"man -w "+$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)
```

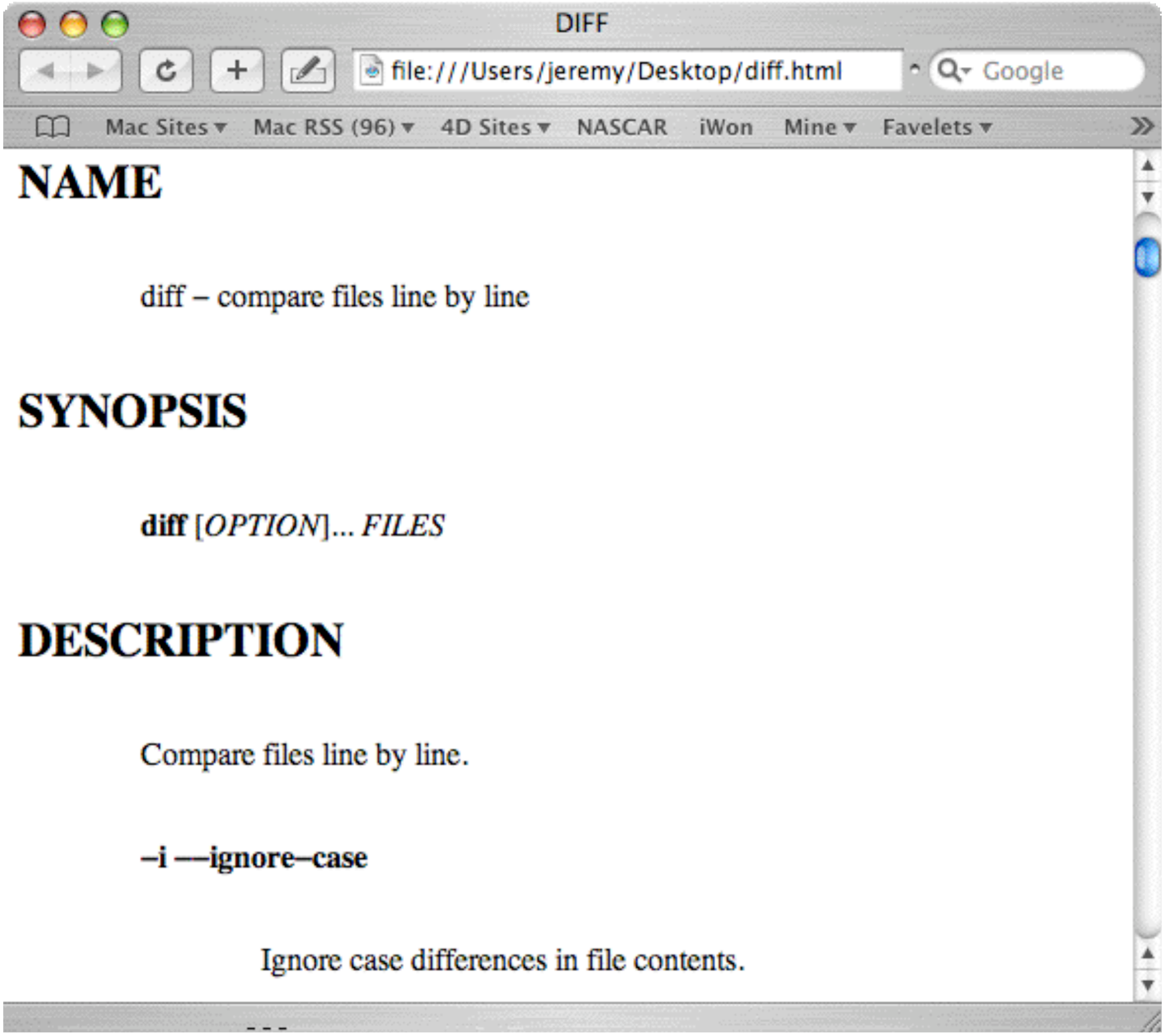
You will notice that in each of these calls /bin/sh was called with an option of -c followed by the commands and options we want to run inside quotes. This causes the bash shell to execute the quoted string. This is a simple trick to get around problems that may arise if you try to pass too complex a command directly through LAUNCH EXTERNAL PROCESS.

In this case the only line that was likely to cause problems is the call to groff. However, it doesn't hurt to explicitly execute commands in the bash shell.

The first line calls the man command with the -w option. This tells the Man command to return the path to the file it would display if it was asked to display the man page.

The next line trims off a line feed that is added when the call is made.
The third line calls the groff utility. The option -Thtml tells groff to format in HTML. The -man tells groff it’s a man page that it will be formatting. Next comes the path to the file it should use as input, in this case the path to the man page that was retrieved previously. The > sign is an instruction to send the output to a file. The last parameter is the path to the file to use for the output.

The last line opens the file in the default application for this file type, most likely a web browser.



NAME

diff – compare files line by line

SYNOPSIS

diff [*OPTION*]... *FILES*

DESCRIPTION

Compare files line by line.

–i —ignore–case

Ignore case differences in file contents.

Summary

This example has given you some ideas about how you can incorporate the built-in commands available for you on each platform. It really only touched the surface on the commands available, particularly on OS X.

Using LAUNCH EXTERNAL PROCESS you can find out about and change the environment your 4D application is running in.

On Mac OS X you can take advantage of utilities such as groff and curl (see TN 05-18), run AppleScripts using osascript, and much more.

On Windows you have access to all of the DOS commands as well as the ability to pass flags to executables that will accept them.

To learn more about what is available on Mac OS X view the complete list of Man pages using the provided example db or an application such as ManOpen (http://www.clindberg.org/projects/ManOpen.html).

A complete list of DOS commands and how to use them is available on Microsoft's web site (http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ntcmds.mspx?mfr=true).