Stringer

Henri Veisterä

COLLABORATORS

	TITLE : Stringer	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Henri Veisterä	February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Stringer

1.1 Stringer AmigaGuide document

Stringer V2.1

1.2 Overview

1.0 Overview

This AmigaGuide® Document is formatted like this:

Section titles are in bold-underline. Section sub-titles are in bold. Commands or options to be typed to CLI are in this color. Sample output from CLI is in this color. The button-row at the bottom of each node duplicates the usual browser commands for easy keyboard browsing.

This document is meant to be displayed with the XHelvetica and XCourier fonts included in the MagicWB package. With other fonts, this document is still readable but the indentation might appear incorrect.

1.3 Requirements

1.1 Requirements

Stringer needs AmigaOS 2.1 (V37) or later and a minimum of 25 kb's of free mem to operate. Stringer is pure and can be made resident.

For searching in archives you need the corresponding unarchiving utility (LhA, UnZip, etc.) in your search path and to display text files you need a text viewer (More, Less, Most, etc.).

1.4 The Story

1.2 The Story

Yes, yet another Search replacement ... with bells on it. As it happens, I wrote this purely for my own usage and amusement and was surprised to see that there were so many similar utils already out there. This one started its life as a tool for counting strings, mainly very short strings (as in counting the relative frequencies of the alphabets) and it just grew from there.

The current version of Stringer, however, is not just a Search replacement. It is a powerful search and replace tool with a large number of options to get the desired results. Currently Stringer has among others the following features:

- Stringer can search for any number of multiple strings

- Strings can be of any size and they can be substrings of other strings searched for and they can be

read from a file

- Search and Replace capability for multiple source and destination strings with optional individual

prompting and backups

- Can search for strings in archive files (.lha, .zip, etc.)

- Can hunt for files using file names or file comments

- Search statistics for matched strings: match counts, relative space usage and relative frequencies

- Can start a pager for immediate viewing of the matched file

- Various options for modifying the output to your needs

- Complete emulation of C:Search with identical ouput if so desired

- Comprehensive documentation with multiple examples and on-line help

- Fastest operation of any utility of this kind (see the test section)

Back to Main Index Back to Overview Index Next Topic

1.5 Installation

1.3 Installation

For an easy installation execute the installation script provided with the archive.

To install Stringer manually:

1. Copy the executable to somewhere in your Path:

Copy Stringer C:

2. Copy one of the provided default preferences to ENV: and ENVARC:

Copy Stringer.prefs ENV:

Copy Stringer.prefs ENVARC:

Back to Main Index Back to Overview Index Next Topic

1.6 Usage

2.0 Usage

Stringer is run from the CLI with the basic command line looking like this:

Stringer SOURCEFILES STRINGS [OPTIONS]

Stringer behaves exactly like Search with its default options so you can immediately use it as you did Search before.

For a quick help about command line options start Stringer from CLI without any options.

Back to Main Index

1.7 Command line options

2.1 Command line options

To see more on an option or for a related example click the More button.

More Option Description

Required options:

FROM/M/A - Source files to search from

SEARCH/A - Strings to search or a file name containing the strings

Separate strings with "," in CLI or "<CR>" in a file

Output modifiers:

N=NONUM/S - No line numbering, faster search

L=NOLINES/S - Don't show lines were a match was found

Q=QUICK/S - Less CR's in the CLI output

O=ONLY/S - Show only names of files where a match was found

ANSI/S - Highlight matched string in the output

FC=FILECOL/K/N - Color for filenames (a pen number from 0 to n)

HC=HITCOL/K/N - Color for a matched string (a pen number from 0 to n)

QU=QUIET/S - No output to CLI, only a result code

Extra information options:

V=VERBOSE/S - Show file size and number and size of strings

T=SHOWTIME/S - Show elapsed time

M=MATCHES/S - Show number of matches in current file

R=REPORT/S - Print out a full report of what was found and where

S=STATS/S - Display search statistics

AS=ALLSTATS/S - Just like STATS but for multiple files

General options:

B=BUFFER/K/N - Work buffer size in kilobytes, defaults to 100 kb

C=CASE/S - Case sensitive search

A=ALL/S - Recursively scan the file tree

I=INVERT/S - Show lines that do not match the search criteria

P=PATTERN/S - SEARCH is an AmigaDOS pattern

F=FILE/S - Hunt for a filename, SEARCH=filename to search for

FN=FILENOTE/S - Hunt for a filenote, SEARCH=filenote to search for

BIN/S - Force Stringer to search through binary files.

MH=MAXHIT/K/N - Give up on a file when MAXHIT matches are found

SF=STRFILE/S - Load strings from a file, SEARCH=filename

NP=NOPREFS/S - Ignore ENV:Stringer.prefs

ARC/S - Also search through archive files (.lha, .zip, etc.)

SHOW/S - Call up a text viewer to view the matched file

H=HELP/K - Displays information on an option

Replace options:

RE=REPLACE/K - Replace matched strings with these strings

ASK/S - Ask before replacing

SAFE/S - Replace to a new file without deleting the old one

OW=OVERWRITE/S - Overwrite in REPLACE

Yup, a lot of command line options. If this seems a bit overwhelming just browse through the command line examples and all will be clear.

Back to Main Index Back to Usage Index Next Topic

1.8 Output control

2.2 Output control

Specifying the L=NOLINES option means that you won't get any output on your screen. So, use it only in combination with the M=MATCHES and/or S=STATS options. Using the QU=QUIET option, you won't get any output at all except the error messages not even the filenames. Using the O=ONLY option displays only the file names where a match was found so, in effect, this is the same as specifying L=NOLINES, M=MATCHES and Q=QUICK, the output, however, looks different in each case.

The options R=REPORT and AS=ALLSTATS are immune to the QU=QUIET option. This is in the likely case you would want to see nothing but the reports.

Use the ANSI option to add some color to your output. You can control the colors with the FC=FILECOL and HC=HITCOL options.

Note that if you use the MH=MAXHIT and S=STATS options together the stats won't be correct since the search is aborted when MH=MAXHIT hits are found.

While Stringer is running you can abort the search with CTRL-C or you can skip the current file with CTRL-D.

Back to Main Index Back to Usage Index Next Topic

1.9 Script file usage

2.3 Script file usage

Stringer exits with these return codes:

0 = One or more match found

5 = No matches found

20 = A fatal error occurred

Use Stringer in a script file like this:

--- AmigaDOS script file ---

Failat 20

Stringer Text.doc PoroJesse QUIET

If warn

Echo "Didn't find PoroJesse in Text.doc."

Else

Echo "Found the string PoroJesse in Text.doc !"

EndIf

--- AmigaDOS script file ---

Back to Main Index Back to Usage Index Next Topic

1.10 Preferences

2.4 Preferences

Stringer reads your preferences from a file called ENV:Stringer.prefs. The format of the file is this: the first line contains your default command line options and the second line contains your default pager and the lines after that are used to handle the different archive formats used.

--- Example ENV:Stringer.prefs file ---

ANSI MAXHIT=200 QUICK BUFFER=400 SAFE

MyPager -l %u "%f"

.pp ppdecrunch "%s" "" COLOR 4 QUIET

.lzh lha -q e "%s"

.lha lha -q e "%s"

.lzx lzx -q x "%s" >NIL:

.zip unzip -o -qq "%s" >NIL:

--- Example ENV:Stringer.prefs file ---

2.4.1 Your default pager

In the second line you set your default pager (a utility that can display text files for you, for example, Less, MuchMore, PPShow, Most, Vinci, AmigaGuide or MultiView). In this line the formatting commands are replaced as follows: %f is replaced with the name of the text file, %s is replaced with the string we found, %ld is replaced with the byte offset where in this text file the string was found for the first time and %u is replaced with the line number where the string was found. So, Pager -b %ld "%f" would result in a command line like Pager -b 24365 "mytext.txt" being executed when Stringer starts your pager for you. Pager -l %u "%f" would result in a similar execution but instead of the byte offset of the match the line number of the match is used.

After Stringer starts your pager for you it will wait until the pager exits. You can change this behaviour with adding a Run command to the start of your default pager command line.

For another approach take a look at the example ARexx scripts provided in Stringers distribution archive for Most and Vinci.

See Command Line Example 11 for how to use this option.

2.4.2 Different archive formats

In the third and the following lines you define the archive formats Stringer can use. Currently Stringer recognizes the .pp .lzh .lha .lzx and .zip formats and calls the PPDecrunch, LhA, LZX and UnZip utilities to unpack these archives. So, these utils have to be in your search path for this option to work. If you want to add more archive formats you have to edit the ENV:Stringer.prefs file. Format for an archive line in this file is:

.suffix<TAB>commandline_to_extract_to_current_dir<CR>

In the command line you give in this file the formatting command %s is replaced with the archive name. Do not use archivers that automatically delete the original packed file after decrunching it (e.g., some versions of uncompress and gzip). If you do, you will lose your archive forever.

You would usually want to redirect the archiver's output to >NIL: but if you want to see the actual unpacking information in your CLI you can leave out the output redirection. Also disable any interactivity your unpacker might have (e.g., asking if you want to replace these files, create directories, etc.).

Command Line Example 10 illustrates further how to use this option.

2.4.3 Setting your preferences

To set your preferences for Stringer edit or create the ENV:Stringer.prefs file with any text editor, which can handle ASCII text.

To make your prefs permanent do this:

Copy ENV:Stringer.prefs ENVARC:Stringer.prefs

Your preferred command line options are read in first and then your command line options are 'added' to these. To temporarily disable your preferred command line options use the command line option NOPREFS.

Back to Main Index Back to Usage Index Next Topic

1.11 Example ARexx scripts

2.5 Example ARexx scripts

Since most pagers (well, almost all of them, shame on you text viewer coders =) don't sport a command line option for jumping directly into a certain place in the text file viewed you can instead of starting the pager directly start an ARexx script. Some pagers have an ARexx port and most of them have an ARexx command for jumping to a specific place in a text file. So, you'll need some ARexx knowledge to make Stringer play ball with these CLI challenged pagers.

These are example scripts on how to use a text viewer's ARexx port to display a text file for you. Take a look at them if you are interested in making your preferred text viewer jump to the correct location in the text file being viewed. There is no need for a separate ARexx script if you don't mind that your text viewer always starts at the beginning of the file.

Example script for Most

Example script for Vinci

Back to Main Index Back to Usage Index Next Topic

1.12 Example command lines

3.0 Example command lines

Back to Main Index

1.13 Example 1 - Vanilla

3.1 Example 1 - Vanilla

Stringer Include/#? ggi_RastPort ALL

This is straight forward stuff ... walk through the Include directories searching for the string ggi_RastPort. The search is not case sensitive. This will produce the same output as using the C: command Search only a bit faster.

You don't have to use the keywords FROM and SEARCH when there is no confusion in the command line about what is an option and what is an argument to an option or if the string searched for is longer than just one character. For example: if you want to search for a string in a file called 'help' you would have to specify FROM help or otherwise you would see Stringers on-line help .

If you want to search for and replace strings which include characters you can't type into the CLI window (like CR's or TAB's) you can insert these characters either in hex or decimal numbers with a backslash (\) character. The string "My\aBString\247"

would be translated into "My«String÷". Two numeric (0-9, A-F, a-f) characters after the backslash mean it is a hex number. Three characters mean it is a decimal number. If you want to search for a string that includes the actual backslash character use two backslashes in a row (\\).

NOTE: In the current version of Stringer you can't search or replace strings which include the characters \000 or \001 since these are used internally by Stringer to mark end of strings and end of buffers.

Back to Main Index Back to Examples Index Next Topic

1.14 Example 2 - Multiple strings

3.2 Example 2 - Multiple strings

Stringer #?new#?.txt "STRING ONE,STRING TWO" QUICK

This example shows the quick way to search for multiple strings from multiple files. The strings include spaces so they are enclosed in quotes. Each string is separated with a comma. If you want to search for a string that includes a comma use two commas in a row: "like I said,, then". The QUICK option means that the name of the file we currently scan is shown but is erased by the next name.

Back to Main Index Back to Examples Index Next Topic

1.15 Example 3 - A quick scan

3.3 Example 3 - A quick scan

Stringer dh0:sc/include/#? ALL SEARCH=select B=400 CASE ONLY M

Here we have a quick scan of all the files in this directory and all its child directories. We scan for the word 'select' and the final output will only show the names of the files where the string was found and a count of matches. Like this:

--- Stringer Output ---

DH0:sc/include/intuition/imageclass.h 2 matches

DH0:sc/include/intuition/intuition.h 22 matches

DH0:sc/include/intuition/preferences.h 2 matches

DH0:sc/include/intuition/screens.h 1 matches

DH0:sc/include/libraries/asl.h 5 matches

DH0:sc/include/libraries/gadtools.h 4 matches

DH0:sc/include/libraries/diskfonttag.h 2 matches

DH0:sc/include/resources/battmembitsamiga.h 2 matches

DH0:sc/include/devices/serial.h 2 matches

DH0:sc/include/devices/hardblocks.h 1 matches

DH0:sc/include/devices/inputevent.h 3 matches

DH0:sc/include/devices/parallel.h 4 matches

DH0:sc/include/graphics/gels.h 2 matches

DH0:sc/include/hardware/cia.h 5 matches

DH0:sc/include/hardware/custom.h 1 matches

--- Stringer Output ---

Back to Main Index Back to Examples Index Next Topic

1.16 Example 4 - Complex patterns

3.4 Example 4 - Complex patterns

Stringer FROM imageclass.h SEARCH=#?0x0[0-9A-F]L#? PATTERN

If you want to search for a complex AmigaDOS pattern use the option PATTERN. Note that there is no sense in using a pattern like SEARCH=(#?keput#?|#?sossut#?) since SEARCH=keput,sossut without the option PATTERN will give the same result and much faster. In this example we look for lines that include an ASCII hex number from 0x00L to 0x0FL. The option CASE applies here also. The pattern is matched against each line in the source file, meaning that a pattern SEARCH=timeout would only match a line that contains nothing but the word timeout.

Here's how to make up an AmigaDOS pattern:

--- illegally/without permission quoted from the autodocs ---

The patterns are fairly extensive, and approximate some of the ability

of Unix/grep "regular expression" patterns. Here are the available

tokens:

? Matches a single character.

Matches the following expression 0 or more times.

(ab|cd) Matches any one of the items separated by '|'.

~ Negates the following expression. It matches all strings

that do not match the expression (aka ~(foo) matches all

strings that are not exactly "foo").

[abc] Character class: matches any of the characters in the class.

a-z Character range (only within character classes).

% Matches 0 characters always (useful in "(foo|bar|%)").

* Synonym for "#?", not available by default in 2.0. Available

as an option that can be turned on.

"Expression" in the above table means either a single character

(ex: "#?"), or an alternation (ex: "#(ab|cd|ef)"), or a character

class (ex: "#[a-zA-Z]").

--- illegally/without permission quoted from the autodocs ---

Back to Main Index Back to Examples Index Next Topic

1.17 Example 5 - Statistics

3.5 Example 5 - Statistics

Stringer Text.txt Strings.txt STRFILE B 300 C V T S L

This is a typical statistical output command line. You get output looking quite different from Search. We go through Text.txt looking for the strings in the file Strings.txt. Strings.txt might look like this:

String1

Stringer2

Str3

So, it's a text file where the strings are separated with CR's. Internal buffer used is 300 kilobytes in size and matching is case sensitive and you'll see the file size and the number of strings in the file Strings.txt and their collective size. You will see something like this:

--- Stringer Output ---

Searching 1801657 bytes of data for 3 strings (total 20 bytes) ...

String Hits All Relat

' String1' 95889 372 964

' Stringer2' 2557 12 25

' Str3' 1001 2 10

TOTAL: 3 strings 99447 386 1000

Time elapsed: 10 secs (7411549 ticks, 709379 ticks/sec)

--- Stringer Output ---

This tells you the following:

String1 occurs 95889 (Hits column) times in the text and 96.4 (Relat column) percent of the matched strings were String1. String1 makes up 37.2 (All column) percent of the whole text.

Stringer2 occurs 2557 times in the text and 2.5 % of the matched strings were String2. Stringer2 takes up approx. 22000 bytes (= 0.012 * 1801657) of the 1801657 bytes in the whole file.

Str3 occurs 1001 times in the text and 1.0 % of the matched strings were Str3.

The TOTAL line gives the added sums of all the above. The whole operation took 10 seconds.

Back to Main Index Back to Examples Index Next Topic

1.18 Example 6 - Hunt files

3.6 Example 6 - Hunt files

Stringer DH0: DH1: DH2: #?.txt ALL FILE

Hunt volumes DH0: , DH1: and DH2: for any files that end in letters .txt . This quickly scans the whole directory structure of the three volumes above and displays any files found matching the search criteria.

Note that you can alternatively hunt for a string in a filenote with the FILENOTE command line option. This works just as the FILE option but instead of file names the filenotes are used for the matching. The FILENOTE option overrides the FILE option. The pattern searched for is a standard AmigaDOS pattern .

Back to Main Index Back to Examples Index Next Topic

1.19 Example 7 - Relative frequencies

3.7 Example 7 - Relative frequencies

Stringer bigtext#?.txt a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z L ALLSTATS >strout.txt

Go through the files bigtext#?.txt and count all the alphabets in these texts. We use the option ALLSTATS since we want to see the combined statistics for all of the multiple files. Output statistics to a file called strout.txt.

Sort FROM strout.txt TO sorted.txt COLSTART 45 NUMERIC

Use the AmigaDOS command Sort and you get output like this (file sorted.txt):

--- Stringer Output ---

String Hits All Relat

’ Z’ 1397 0 1

’ Q’ 1425 0 1

’ X’ 3849 2 3

’ J’ 5831 3 5

’ V’ 9807 6 8

’ B’ 17773 10 16

’ P’ 19669 12 17

’ W’ 19748 12 17

’ F’ 21239 13 19

’ K’ 21733 13 19

’ G’ 22208 13 20

’ Y’ 23191 14 20

’ M’ 32917 20 29

’ C’ 35719 21 32

’ L’ 38968 23 35

’ U’ 39221 24 35

’ D’ 40567 24 36

’ H’ 48059 29 43

’ R’ 64312 39 57

’ S’ 70609 43 63

’ N’ 77408 47 69

’ I’ 80250 49 72

’ O’ 84288 51 76

’ T’ 99484 61 89

’ A’ 102745 63 92

’ E’ 126617 77 114

TOTAL: 26 strings 1109034 669 1000

--- Stringer Output ---

If the files bigtext#?.txt were ASCII text files in English you can see from the above the relative frequencies of the alphabets in the English language (or a good approximation of it). There is a 6.9 percent probability that the next letter you see is the letter ’N’. The most common letter is the letter ’E’.

Back to Main Index Back to Examples Index Next Topic

1.20 Example 8 - Report card

3.8 Example 8 - Report card

Stringer DH0:Pics/#? JFIF,GIF,FORM CASE QUIET MAXHIT=1 BIN REPORT B=2

Scan the directory DH0:Pics and report which files were JPEG, GIF or IFF files. The option MAXHIT=1 makes the search much faster since Stringer can jump to the next file immediately after the identifying string was found. Also, the option BIN is required since we are searching through binary files. We assume that all the files in this directory are JPEG, GIF or IFF files and therefore the small buffer size (2 kb) actually speeds up the search considerably.

Note that you can use the option M=MATCHES together with the REPORT option. This way you can see each file where a string was found and how many times it was found in each of those files.

Back to Main Index Back to Examples Index Next Topic

1.21 Example 9 - Search and Replace

3.9 Example 9 - Search and Replace

Stringer DH0:Text/Documents/#? "Jim Henson" ASK REPLACE "Frank Oz"

Search all the documents in this dir for the name "Jim Henson" and replace it with the name "Frank Oz". Every time the string "Jim Henson" is found you are prompted with Replace (Y[es]/n[o]/a[ll]) ?. Pressing just return or typing 'y' or 'yes' the replace is done. Typing 'n' or 'no' means that this particular string is not replaced. Typing 'a' or 'all' means that you won't see this prompt again and every string found onwards from here is automatically replaced.

You can search and replace multiple strings with the same command line syntax as is used when searching for multiple strings:

Stringer DH0:Text/Documents/#? boy,girl CASE REPLACE woman,man

Again, search all the documents and replace every instance of boy with woman and every girl with man. The replace is case sensitive. This time you are prompted for each individual file where we are about to replace something rather than for each individual string. You can disable this prompting with the QUIET option.

If you want to preserve the file size when your replacement string is longer than the original string, use the option OVERWRITE:

Stringer #? SEARCH topaz.font REPLACE thintopaz.font OVERWRITE

Every time the string "topaz.font" is found it is overwritten with the string "thintopaz.font". For example: FONTtopaz.font@@@@@@@B@ would change to FONTthintopaz.font@@@@A B@ .

Note that the new file is built up in the same directory as the original so there must be enough space on this device for the replace operation to work.

You can't use the REPLACE option together with the PATTERN or INVERT options since MatchPattern() doesn't report where the string was found and in the case of invert it is not clear what to replace.

If you are conserned about losing an important document the option SAFE will preserve the original document with a suffix .bak added to its name. This option is by default on and you have to edit the ENV:Stringer.prefs file to turn it off.

In case a disk error happens in the middle of a replace operation the workfile Stringer uses (your_original_name.strire) might be left in the original directory. This workfile is not deleted when a disk error occurs so that you will have at least this copy left of your text file. In such a case Stringer exits with an error message explaining the nature of the error.

Back to Main Index Back to Examples Index Next Topic

1.22 Example 10 - Hitting the Archives

3.10 Example 10 - Hitting the Archives

Stringer #? myname ARC ALL

This is just a simple search like example 1 but in this case if a file is found to be an archive file it is first unpacked and the files within it are also included in the search. Each file is searched for the string 'myname' and the matches are reported accordingly.

If the archives contain binary files you want to search through remember to include the BIN option in the command line.

Stringer also unpacks any nested archives within other archives and searches all the files in these too.

Note that the unpacking is done into your T: assign so this has to be assigned and this device has to have enough free space for this operation to work.

Currently Stringer recognizes the .pp .lzh .lha .lzx and .zip formats and calls the PPDecrunch, LhA, LZX and UnZip utilities to unpack these archives. You can add more archive formats for Stringer in the ENV:Stringer.prefs file. The ARC option will not work if the ENV:Stringer.prefs file is not correctly installed.

Back to Main Index Back to Examples Index Next Topic

1.23 Example 11 - Just Paging through

3.11 Example 11 - Just Paging through

Stringer #?.txt HappyJoy C SHOW

Invokes a text viewer for the first file that contains the exact string 'HappyJoy'. The default pager is More. You can set your default pager and the command line used in the ENV:Stringer.prefs file. If the SHOW option doesn't work, check that your command line in the ENV:Stringer.prefs file is correct.

You can view a text file inside an archive using the ARC option together with the SHOW option but if you Run your text viewer or your text viewer detaches itself from the CLI this will leave you with the temporary directory Stringer used in the unpacking operation and this one unpacked file. The temporary directory will be of the form T:Stringer_xxxx/. Stringer will complain that it couldn't delete this file. You will have to decide what to do with this one file afterwards.

Back to Main Index Back to Examples Index Next Topic

1.24 Example 12 - Help me !

3.12 Example 12 - Help me !

Stringer offers you on-line help on any of the command line options:

Stringer HELP REPLACE

This would display a short help text on the REPLACE option. The help text will look like this:

Option: REPLACE Abbreviation: RE Type: KEYWORD

Info: Replace matched strings with these strings

This tells you the options name and the command line abbreviation and type. Type KEYWORD means that this option needs an additional argument. In this case the string or strings we would use in the replace operation. Other possible types are REQUIRED and SWITCH. REQUIRED is an option that has to be in the command line for Stringer to work. SWITCH is, as the name suggests, a command line switch. If the switch is found in the command line it is on and otherwise it is off.

You don't have to give the whole option name as an argument to HELP. The first few letters will be enough. In this example just typing Stringer HELP REPL would be enough to distinguish REPLACE from REPORT.

Back to Main Index Back to Examples Index Next Topic

1.25 The test section

4.0 The test section

I spent a day compiling this stuff, so you better read it through. A quick summary can be found at the end of this section. In all the tests I experimented with different command line options to get the best results, any output was redirected to NIL:. Since some of the utils don't support searching multiple files, multiple directories or multiple strings script files were created for these utils where they were just executed a multiple of times.

Test hardware:

68030/28 Mhz, 4 MB/32 bit Mem, Seagate 501 MB SCSI-2 HD, OS3.1

Test software:

Search 40.1 by AmigaDOS

ssearch 1.4 by Stefan Sticht

FSearch 1.2 by Edwin H. Bielawski

ZSearch 1.0c by Alessandro Zummo, 030 version used

FlashFind 1.2 by Frank Würkner

Stringer 1.1 by Henri Veisterä

1.26 Test 1 - Algorithm test

4.1 Test 1 - Algorithm test

Source file is 1801657 bytes of ASCII text with variable line lengths. Search one string (strlen = 7 bytes) case sensitive from RamDisk. The string is found two times in the source file.

Util Time/s RelSpeed

search 92.9 84.5

fsearch 15.3 13.9

ssearch 11.1 10.1

zsearch 3.7 3.4

flashfind 1.1 1.0

stringer 1.1 1.0

This is a test of just the search algorithms and file access times are minimal here. FlashFind loses by a few hundredths of a second.

Back to Main Index Back to Test Index Next Topic

1.27 Test 2 - Multiple small file test

4.2 Test 2 - Multiple small file test

Source files are the Commodore assembler include files 218 files - 16 dirs - 871692 bytes. Searching for one string 11 bytes in size which is found three times in one of the source files. Search is case sensitive from a hard drive.

Util Time/s RelSpeed

search 99.2 4.3

fsearch 57.1 2.5

zsearch n/a,47.1 2.0

ssearch 41.9 1.8

flashfind 35.9 1.6

stringer 23.0 1.0

Many small files (average file size ~4000 bytes) to go through so the importance of the search algorithm is less here, a large portion of the time is spent loading the files. Stringer wins through efficient multiple file implementation. ZSearch can't search multiple files so the time here is from a script file created for this test.

Back to Main Index Back to Test Index Next Topic

1.28 Test 3 - Multiple large file test

4.3 Test 3 - Multiple large file test

Commodore AutoDocs 30 files - 1768252 bytes searching for one string (7 bytes). String is found 47 times in the source files. Search is case sensitive from a hard drive.

Util Time/s RelSpeed

search 159.8 24.2

ssearch 63.4 9.6

fsearch 63.0 9.5

flashfind 52.7 8.0

zsearch n/a,14.4 2.2

stringer 6.6 1.0

Stringer really shines through here. Easily the fastest 8-P.

Back to Main Index Back to Test Index Next Topic

1.29 Test 4 - The really unfair test =)

4.4 Test 4 - The really unfair test =)

Commodore AutoDocs 30 files - 1768252 bytes. Searching for three strings (20 bytes). Strings are found 375 times in the source files. Search is not case sensitive and from a hard drive.

search 554.2 38.5

fsearch n/a,202.6 14.1

ssearch n/a,195.7 13.6

flashfind 353.1,159.1 11.0

zsearch n/a,43.9 3.0

stringer 14.4 1.0

As you can see, this is why I made Stringer in the first place so it should show here ... and it does :-! Only Stringer, FlashFind and Search are able to search for multiple strings. All the results after the commas are from script files created for this test.

Back to Main Index Back to Test Index Next Topic

1.30 Grande Test Total

4.5 Grande Test Total

Test1 Test2 Test3 Test4 Total

Time/s Rel Time/s Rel Time/s Rel Time/s Rel Ave Rel

search 92.9 84.5 99.2 4.3 159.8 24.2 554.2 38.5 37.9

fsearch 15.3 13.9 57.1 2.5 63.0 9.6 202.6+ 14.1 10.0

ssearch 11.1 10.1 41.9 2.0 63.4 9.5 195.7+ 13.6 8.8

flashfind 1.1 1.0 35.9 1.8 52.7 8.0 159.1+ 11.0 5.5

zsearch 3.7 3.4 47.1 1.6 14.4+ 2.2 43.9+ 3.0 2.6

stringer 1.1 1.0* 23.0 1.0* 6.6 1.0* 14.4 1.0* 1.0

A * marks the best result, a + means that the util didn't support this test and a script file was used instead.

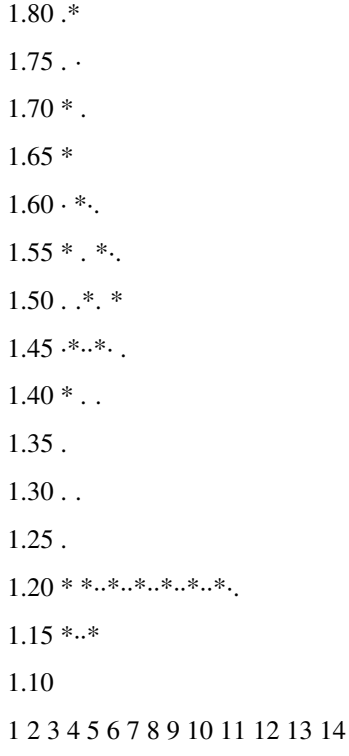
So, twice as fast as the nearest contender ... but since zsearch is really limited in its uses I could say 6 to 38 times as fast as the rest! Most pleasing for me is that Stringer is at its best at just the kind of work I use a string searcher for (tests 3 and 4). Curiously enough, the only util whose search algorithm is more powerful than Stringers (FlashFind is faster when strlen > 7, and the source file is > 2 MB) fails totally in real world performance at least on my machine. I know that FlashFind performs better on other peoples Amigas. I suspect this has to do with faulty a-sync IO routines.

Back to Main Index Back to Test Index Next Topic

1.31 Buffer efficiency graph

4.6 Buffer efficiency graph

No test section would be complete without a beautiful graph. Thus, for your enjoyment: the following graph represents the time spent searching for one string from a 1759 kb source file with varying buffer sizes.



On the y-axis is the elapsed time in seconds and on the x-axis is the buffer size used in 100 kilobytes. The asterisks '*' are measured data points and the points '·' in between were interpolated.

From this you can see that after 1200 kb's the usefulness of a greater buffer size is dubious. Note that the low points are situated at 400 kb intervals due to exec.librarys internal optimizations. So, in other words, if you want to use a greater default buffer size use 400, 800 or 1200 kb's.

Back to Main Index Back to Test Index Next Topic

1.32 Author and support

5.0 Author and support

Stringer was written in 100% Assembler by me, Henri Veisterä. For bug reports, fan or hate-mail or general chitchat you can contact me through the net via: hveister@snakemail.hut.fi . Stringer is Freeware. Give it away. Don't sell it. Latest version will always be posted to AmiNet.

Thanks go to Jim Greenidge for beta testing and ideas.

Back to Main Index

1.33 Version history

6.0 Version history

6.1 Version 1.0 - 26-Jun-93

- Ancient stats shower

6.1 Version 1.1 - 16-Aug-96

- First public release

6.2 Version 1.2 - 16-Sep-96

- Internal version never released to the public

- Added the REPORT, ALLSTATS, REPLACE and ASK options.

- Made it so that the option MAXHIT is not ignored when the option NOLINES is on.

- Fixed a bug with multiple files where in some cases the option NOLINES was turned

on for all the files after the first one was done with.

6.3 Version 1.3 - 10-Oct-96

- Internal version never released to the public

- Added the SAFE, ARC, FILENOTE and SHOW options

- Made the REPLACE work with multiple replace strings

- Made the REPLACE option much faster

- Fixed a bug where if the string to be replaced was found many times in one line sometimes

only the first occurrence was replaced.

- Fixed a bug where the very first file examined was not included in the FILE hunt.

- Fixed the display of very long matched lines so that the matched string is always seen.

- Reformatted the docs to an AmigaGuide® Document

6.4 Version 2.0 - 16-Oct-96

- Second public release

- Added the OVERWRITE and HELP options

- Changed REPLACE to always ask if QUIET is not specified

- SHOWTIME now shows elapsed time for whole operation instead of individual files.

- Shows elapsed time for BREAK'ed serches

- Fixed a bug in SHOW where some file handles were left UnLock()'ed

6.5 Version 2.1 - 16-Nov-96

- Third public release

- Added the ability to insert hex or decimal numbers into search and replace strings

- Changed the NONUMS option to NONUM for complete Search emulation

- Fixed a bug where if the CASE and QUIET options were used together with the REPLACE option

the replace was not done.

Back to Main Index

1.34 To do

7.0 To do

These future options are in a descending probability order.

Make archive unpacking BREAK'able from Stringer. Currently you have to Break an unarchiving process from another CLI.

Add options to specify how many lines to display 'around' the one matched line.

Add simple boolean operators for multiple search strings.

Implement internal wildcards instead of using MatchPattern().

Stringer currently has 34 command line options. This is close or over a useful limit for an CLI utility and the casual user might feel swamped by them all. So, I'm planning to do a graphical front-end for Stringer using MUI. This will be just an add-on and will call Stringer to do its actual work but will probably be most helpful for people who might use a search tool occasionaly.

If you have an idea that you think might be useful for other people too, don't hesitate to contact me .

Back to Main Index