User's Guide

CodeSuite-AC Version 1.0

a product of

S.A.F.E.

Software Analysis & Forensic Engineering Corporation

Table of Contents

CodeSuite-AC

CodeSuite-AC® is the academic version of the CodeSuite® collection of computer code analysis tools. The individual tools that comprise the suite of tools include CodeMatch®, FileCount™, and FileIdentify™, all of which are described below.

CodeMatch

CodeMatch compares thousands of source code files in multiple directories and subdirectories to determine which files are the most highly correlated. This can be used to significantly speed up the work of finding source code plagiarism, because it can direct the examiner to look closely at a small amount of code in a handful of files rather than thousands of combinations. CodeMatch is also useful for finding open source code within proprietary code, determining common authorship of two different programs, and discovering common, standard algorithms within different programs.

CodeMatch compares every file in one directory with every file in another directory, including all subdirectories if requested. CodeMatch produces an HTML basic report that lists the most highly correlated pairs of files. You can click on any particular pair listed in the HTML basic report see an HTML detailed report that shows the specific items in the files (statements, comments, strings, identifiers, or instruction sequences) that caused the high correlation.

CodeMatch uses unique algorithms to find various different ways that source code files are correlated. These algorithms can find directly copied source code and even source code that has been modified to avoid detection.

FileCount

FileCount is a utility that counts the number of files, non-blank lines, and bytes in a large set of files in a directory tree. FileCount is useful when using CodeDiff to generate statistics about a set of source code files.

FileIdentify

FileIdentify is a utility that examines all of the file types in a given directory, or an entire directory tree, and reports the associated programming languages if known.

Copyrights, Trademarks, Patents

Copyrights

Trademarks

Patents

CodeSuite-AC is covered by U.S. patents 7,503,035, 7,823,127, 8,255,885, 8,261,237, 8,495,586, 9,003,366, 9,043,375, and 9,053,296.

Using CodeSuite-AC

System Requirements

CodeSuite-AC will run on any computer using any of the following versions of the Microsoft Windows operating system:

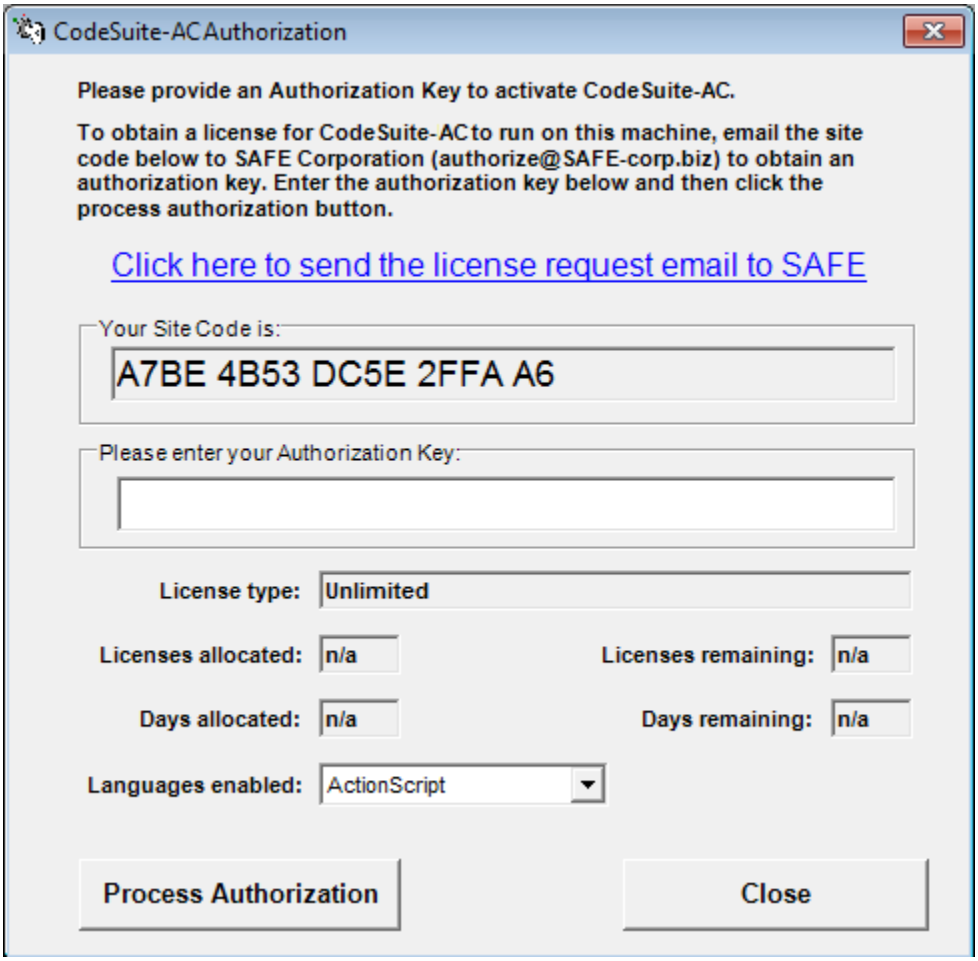
- Windows Vista
- Windows 7
- Windows 8
- Windows 10

Note that CodeSuite-AC will not run on a virtual system and may not run on some systems using a remote desktop.

Licenses

Licenses must be purchased from SAFE Corporation. The FileCount, FileIdentify, and FileIsolate functions of CodeSuite-AC do not require a license.

To request licenses, open the authorization form shown below from the Help menu. Send the site code to SAFE Corporation and the number of licenses requested, along with appropriate payment. SAFE Corporation will send back an Authorization Key that must be entered into the field in the form. Press the process authorization button and the form will show the following information. Licenses are enabled for only one PC and cannot be transferred to another PC.



License Type

The license can be one of three types.

- **File size based.** Used to examine a fixed amount of bytes of source code. Licenses are used up as source code is examined. SourceDetective searches of the Internet also use up licenses.
- **Time based.** Used to examine any amount of code for a fixed number of days. Note that there is still a limit to the number of SourceDetective searches of the Internet that can be performed. If that limit is reached, no more searching can be done for the remainder of the license term unless a new license is purchased.
- **Unlimited.** There is no limit on the number of megabytes that can be examined and there is no expiration date.

Licenses Allocated and Licenses Remaining

These fields indicate the number of licenses that were originally allocated and how many unused licenses remain. These fields are valid only for a megabyte-based license. For other licenses, the fields are not applicable ("n/a").

Days Allocated and Days Remaining

These fields indicate the number of days that were originally allocated for the license and how many days remain on the license. These fields are valid only for a time-based license. For other licenses, the fields are not applicable ("n/a").

Languages Enabled

This pulldown list shows all of the programming languages that are enabled for analysis by the license.

See the SAFE Corporation website for license costs, as they may change.

The Toolbar

The CodeSuite-AC toolbar is shown below.

CodeMatch

This menu selection brings up the CodeMatch form. See the section entitled Running CodeMatch for more information.

FileCount

This menu selection brings up the FileCount form. See the section entitled Running FileCount for more information.

FileIdentify

This menu selection brings up the FileIdentify form. See the section entitled Running FileIdentify for more information.

Help

This menu selection brings up this user's guide.

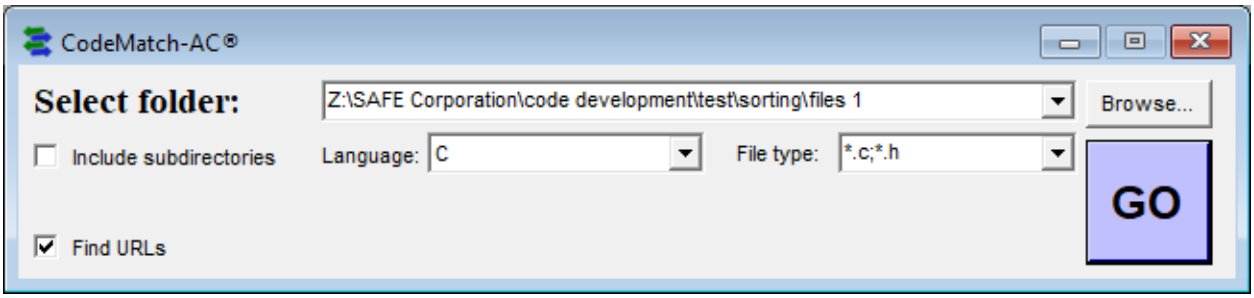
Authorize

This menu selection brings up the authorization form for entering licenses to enable the various tools. See the section entitled Licenses for more information.

CodeMatch

Running CodeMatch

CodeMatch compares files using a set of algorithms to determine their correlation. Below is a screen shot of the CodeMatch form. Following that are step-by-step instructions for running CodeMatch.



Step 1

Select the folder containing source code files for comparison by clicking on the browse button or entering the path in the text field. Check the box to include files in all subdirectories, if desired.

Step 2

Select a source code language from the pulldown menu.

Step 3

Select the files types to compare from the pulldown menu. You can type over the suggested file types with your own file types. Separate multiple file types with a semicolon. Use the * and ? wildcard characters if needed.

Step 4

Check the box to use SourceDetective® to find the URLs where the source code elements can be found.

Step 5

Click on the GO button. The number of licenses, if any, that are required for this run of CodeMatch will be shown. You will have the ability to cancel the CodeMatch run at this point without using up licenses.

You will be then asked for the name of the file and folder to contain the HTML reports.

Resulting HTML reports

After the comparison, HTML reports will be generated. The basic report shows file pairs and their correlation scores. By clicking on a score, a detailed HTML report will come up for that file pair. These detailed reports are kept in subfolders. The detailed reports give more information about how the score was determined, showing specific similarities or differences between the files. The file names are given at the top of the report and include hyperlinks that, when clicked, allow the file to be brought up in a viewer or editor. The back and next buttons on the detailed reports allow you to navigate the detailed reports without going back to the basic report.

The basic report also includes the top websites and URLs where code elements were found online. This can help determine whether code was copied from an online source.

For examples of the reports, see the sections entitled CodeMatch Basic Report and CodeMatch Detailed Report.

CodeMatch Algorithms

The Algorithms

CodeMatch uses several algorithms to determine similarity between two source code files. These algorithms are described below. When multiple files are compared, each match is given a weight and all weights are combined into a single matching score called the correlation score. The file pairs are then ranked by correlation score so that you can examine the most similar files.

Statement matching

CodeMatch looks for identical program statements (i.e., functional source code), ignoring whitespace and eliminating comments and strings. Statements that contain only programming language keywords are not considered matching. For statements to be considered matches, they must contain at least one identifier (non-keyword) such as a variable name or function name.

Comment/string matching

CodeMatch looks for identical comments and strings, ignoring whitespace. Comment lines and strings that contain only programming language keywords are still considered matches.

Instruction sequence matching

CodeMatch looks for sequences of instructions that match. CodeMatch notes the longest such sequence in each pair of files. A sequence matches if the initial programming language statement on each line is identical, regardless of what follows it. Even if variable names are altered in one file, CodeMatch will report similarities in the files. The following shows an example of two identical instruction sequences in C:

```
// File 1
if (x == 5)
{
    // Loop on j here
    for (j = 0; j < Index; j++)
        printf("x = %i", j);
}
else
    break;  // Here's the break

// File 2
if (xyz < 2)
    for (jjj = 0; jjj < i; jjj++)
    {
```

```
            printf("Hello world\n");
        }
    else
        break;
```

Identifier matching

CodeMatch finds every instance in each file where identifiers match exactly. It eliminates programming language keywords and only reports matches for non-keyword identifiers such as variable names and function names.

CodeMatch also finds every instance where an identifier in one file is part of a larger identifier in the other file. For example, the variable name "Index" in one file would partially match the variable names "NewIndex" and "Index1" in the other file. CodeMatch eliminates programming language keywords and only reports matches for non-keyword identifiers such as variable names and function names.

Correlation Score

CodeMatch produces a total correlation score based on the combination of above algorithms that the user chooses when running CodeMatch. The minimum score is 0 while the maximum score is 100.

S.A.F.E.

Code Match

CodeMatch Basic Report
Version: 5.7.2 | Date: 07/24/18 | Time: 23:02:59

SETTINGS | RESULTS | UNCOMPARED FILES | URLS | TOTALS

SETTINGS

Compare files in folder	Z:\SAFE Corporation\code development\test\sorting\files 1 *Not including subdirectories*
File types	*.c;*.h
Programming language	C
To files in folder	Z:\SAFE Corporation\code development\test\sorting\files 1 *Not including subdirectories*
File types	*.c;*.h
Programming language	C
Algorithms selected	• Statement Matching • Comment/String Matching • Identifier Matching • Instruction Sequence Matching • List sequences
Reporting file threshold	4 files

RESULTS

Z:\SAFE Corporation\code development\test\sorting\files 1\aaa.c

Z:\SAFE Corporation\code development\test\sorting\files 1\aaa.c

Score	Compared to file
82	Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_case.c
82	Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_whitespace.c

Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_case.c

Score	Compared to file
82	Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_whitespace.c

Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_with_comments.c

Score	Compared to file
77	Z:\SAFE Corporation\code development\test\sorting\files 1\abc_with_comments.c
71	Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_with_comments.c
71	Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_with_comments.c

Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_whitespace.c

Score	Compared to file
71	Z:\SAFE Corporation\code development\test\sorting\files 1\aaa_with_comments.c
65	Z:\SAFE Corporation\code development\test\sorting\files 1\abc_with_comments.c
65	Z:\SAFE Corporation\code development\test\sorting\files 1\abc_with_comments.c
65	Z:\SAFE Corporation\code development\test\sorting\files 1\abc_with_comments.c
56	Z:\SAFE Corporation\code development\test\sorting\files 1\bpf_image.c
56	Z:\SAFE Corporation\code development\test\sorting\files 1\bpf_image2.c

URLs

Hits	Website
92	libseccomp/scmp_bpf_disasm.c at master · seccomp ...
88	libpcap/bpf_image.c at master · the-tcpdump-group/libpcap ...
86	libpcap: bpf/net/bpf_filter.c Source File - doxygen ...
60	bpf_filter.c - Apple Inc.
54	bpf(4) - Berkeley Packet Filter - GSP Services
38	Description - man pages section 7: Device and Network ...
17	STRING: functional protein association networks
17	String | Definition of String by Merriam-Webster
17	String Class (System) - msdn.microsoft.com
17	String (Java Platform SE 7) - Oracle Help Center

TOTALS

Total number of bytes in files in folder 1 = 23685
Total run time = 2 Seconds

TOP

S.A.F.E.

Code Match

CodeMatch Detailed Report
Version: 5.3.1 | Date: 08/28/08 | Time: 11:33:11

SETTINGS

Compare file 1:	C:\test\C\files 1\bpf_image.c
To file 2:	C:\test\C\files 2\.svn\bpf_image.c
Links to results:	Matching Statements Matching Comments and Strings Matching Instruction Sequences Matching Identifiers Partially Matching Identifiers Score

RESULTS

Matching Statements		
File1 Line#	**File2 Line#**	**Statement**
22	22	#include <windows.h>
23	23	#include <sys/types.h>
35	35	char *fmt, *op
36	36	static char image[256]
37	37	char operand[64]
39	39	v = p->k

40	40	switch (p->code) {
199 204 209 214	199	case BPF_ALU|BPF_OR|BPF_X:
254	254 259 264 269 270	case BPF_ALU|BPF_NEG:

TOP

Matching Comments and Strings

File1 Line#	File2 Line#	Comment/String
2	2	* Copyright (c) 1990, 1991, 1992, 1994, 1995, 1996
3	3	* The Regents of the University of California. All rights reserved.
5	5	* Redistribution and use in source and binary forms, with or without
6	6	* modification, are permitted provided that: (1) source code distributions
7	7	* retain the above copyright notice and this paragraph in its entirety, (2)
8	8	* distributions including binary code include the above copyright notice and
9	9	* this paragraph in its entirety in the documentation or other materials
10	10	* provided with the distribution, and (3) all advertising materials mentioning
11	11	* features or use of this software display the following acknowledgement:

TOP

Matching Instruction Sequences		
File1 Line#	**File2 Line#**	**Number of matching instructions**
22	22	202
43	129	71
46	51	64
46	56	60
46	61	56
46	66	52
46	71	48
46	76	44
46	81	40
46	86	36
46	91	32

TOP

Matching Identifiers							
256	64	BPF_A	BPF_ABS	BPF_ADD	BPF_ALU	BPF_AND	BPF_B
BPF_CLASS	BPF_DIV	BPF_H	bpf_image	BPF_IMM	BPF_IND	bpf_insn	BPF_JA
BPF_JEQ	BPF_JGE	BPF_JGT	BPF_JMP	BPF_JSET	BPF_K	BPF_LD	BPF_LDX
BPF_LEN	BPF_LSH	BPF_MEM	BPF_MISC	BPF_MSH	BPF_MUL	BPF_NEG	BPF_OP
BPF_OR	BPF_RET	BPF_RSH	BPF_ST	BPF_STX	BPF_SUB	BPF_TAX	BPF_TXA
BPF_W	BPF_X	code	fmt	image	INT	jf	jt

op	operand	stdio	string	sys	types	windows	

TOP

Partially Matching Identifiers

File1 Identifiers

0x00FF	BPF_ALU	bpf_filter	BPF_IMM	BPF_IND	BPF_LEN	BPF_MEMWORDS	BPF_RET
BPF_ST	BPF_SUB	EXTRACT_LONG	INT	netlong	types	UCHAR	W32N_htonl
winsock							

File2 Identifiers

0x00004	0x00005	__stdcall	_TEXT	_W32N_ADA	_WAdapter0	_WAdapter1	_WAdapter2
dwDataLen	DWORD	dwType	ERR_IMPLIED	ERR_SUCCESS	H_LOCAL	hAdapter	hClassNet
KEY_READ	LONG	pAdapterInfo	PCHAR	PW_ADAPTER	QueryValue	TChar	VER_WIN32
W0Adapter	W0Window	W0Windows	W32N_Adapt	W32N_NET	WINCARDS	wsprintf	

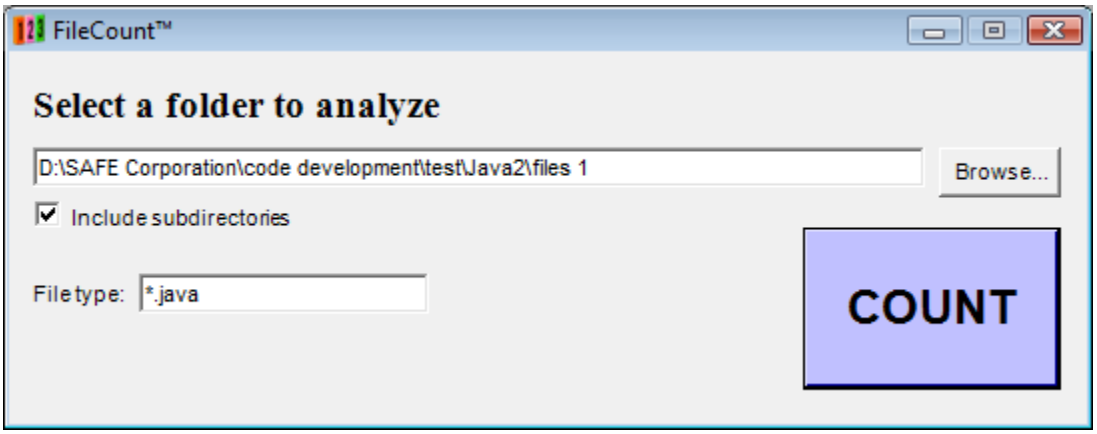
TOP

SCORE 100

FileCount

Running FileCount

FileCount is a utility that counts the number of files, non-blank lines, and bytes in a large set of files in a directory tree. FileCount is useful when using CodeDiff to generate statistics about a set of source code files.



Step 1

Select the folder where the files are that need to be counted by clicking on the browse button or entering the path in the text field. Check the box to include all subdirectories.

Step 2

Type in the file types. Separate different file types with a semicolon. Use the * and ? wildcard characters if needed.

Step 3

Press the count button. FileCount will then search the directory and all subdirectories, if specified, counting all of the files that meet the file type, and counting the total number of non-blank lines and bytes. When complete, a dialog box will appear with these counts.

FileIdentify

Running FileIdentify

FileIdentify allows a directory or directory tree and lists all of the file types found, based on the file name extensions. It also reports all known programming language files based on the file types. Below is a screen shot of the FileIdentify form and step-by-step instructions for running FileIdentify.

Step 1

Select the folder where the files are located that you want to analyze. Check the box to include all subdirectories if you want to analyze files in the subfolders also.

Step 2

Press the go button. You will be asked for the file name and location for a spreadsheet showing all file types and their associated programming languages, if known. FileIdentify will then search the directory and all subdirectories, if specified.

Below is an example of a spreadsheet created by FileIdentify.

	A	B	C
1	Analysis of Extensions		
2	Analysis date	12/16/2012	
3	Folder	Z:\SAFE Corporation\code development\test\FileIdentify\top	
4	Include subfolders	Yes	
5			
6	Files with no extension	0	
7	Files with an empty extension	0	
8	Folder paths too long	0	
9	File paths too long	0	

10			
11	File types	Number of files	Language (if known)
12	.as	37	ActionScript
13	.c	81	C
14	.cdb	6	
15	.csf	1	
16	.flr	1	
17	.gif	47	
18	.htm	181	
19	.jpg	8	
20	.js	413	JavaScript
21	.mako	1	
22	.php	8	PHP
23	.png	49	
24	.swf	517	
25	.txt	66	

The top line shows that the spreadsheet was an analysis of file extensions created by FileIdentify. The second line shows the date that the analysis was run. The third shows the folder name. The fourth line indicates whether or not subfolders were included in the analysis.

Line 6 gives the number of files that had no extension while line 7 gives the number of files that had an empty extension, meaning the file name ended in a dot. Line 8 gives the number of folders that exceeded the maximum number of characters and could thus not be examined while line 9 gives the number of file paths, meaning the folder name plus the file name, that exceeded the maximum number of characters and could thus not be examined.

Lines 12 through 25 show the files types that were found, in column A, the number of files for each file type, in column B, and the programming language, if known, in column C.

Languages

Languages Supported

The following programming languages are currently supported:

ABAP	ASM-6502	ASM-65C02	ASM-65816	ASM-M68k
BASIC	C	C++	C#	COBOL
Delphi	DRI ASM	Flash ActionScript	Fortran	FoxPro
Go	Java	JavaScript	Kotlin	LISP
LotusScript	MASM	MATLAB	MPE/iX	Objective-C
OpenEdge	Pascal	Perl	PHP	PL/M
PowerBuilder	PowerHouse	Progress	Prolog	Python
RealBasic	Ruby	Scala	SQL	Swift
TAL	TCL	Verilog	VHDL	Visual Basic

Check the SAFE Corporation website for new language modules, available at no charge, as they become available. If the language you need is not available, contact SAFE Corporation about creating it for a nominal fee.

Contacting SAFE Corporation

Contacting SAFE Corporation

Software Analysis and Forensic Engineering Corporation
20863 Stevens Creek Blvd.
Suite 456
Cupertino, CA 95014
www.SAFE-corp.com

Tel. (408) 517-1167
Fax (408) 693-3727
Email: Support@SAFE-corp.com

Index