C Shell Script CodeCount™

Counting Standard

University of Southern California

Center for Systems and Software Engineering

April , 2010

Revision Sheet

Date	Version	Revision Description	Author
5/12/2010	1.0	Original release	CSSE
1/2/2013	1.1	Updated document template	CSSE

Table of Contents

1. Definitions

1.1. **SLOC –** Source Lines of Code is a unit used to measure the size of software program. SLOC counts the program source code based on a certain set of rules. SLOC is a key input for estimating project effort and is also used to calculate productivity and other measurements.

1.2. **Physical SLOC –** One physical SLOC is corresponding to one line starting with the first character and ending by a carriage return or an end-of-file marker of the same line, and which excludes the blank and comment line.

1.3. **Logical SLOC –** Lines of code intended to measure "statements", which normally terminate by a semicolon (C/C++, Java, C#) or a carriage return (VB, Assembly), etc. Logical SLOC are not sensitive to format and style conventions, but they are language-dependent.

1.4. **Data declaration line or data line –** A line that contains declaration of data and used by an assembler or compiler to interpret other elements of the program.

1.5. **Compiler Directives –** A statement that tells the compiler how to compile a program, but not what to compile. C Shell Script does not contain any compiler directives.

1.6. **Blank Line –** A physical line of code, which contains any number of white space characters (spaces, tabs, form feed, carriage return, line feed, or their derivatives).

1.7. **Comment Line –** A comment is defined as a string of zero or more characters that follow language-specific comment delimiter.

C Shell Script comment delimiter is "#". A whole comment line may span one line and does not contain any compliable source code. An embedded comment can co-exist with compliable source code on the same physical line. Banners and empty comments are treated as types of comments.

1.8. **Executable Line of code –** A line that contains software instruction executed during runtime and on which a breakpoint can be set in a debugging tool. An instruction can be stated in a simple or compound form.

- An executable line of code may contain the following program control statements:
  - Selection statements (if, switch, case)
  - Iteration statements (foreach, while)
  - Empty statements (one or more ";")
  - Jump statements (goto, break, continue, exit)
  - Expression statements (function calls, assignment statements, operations, etc.)
  - Block statements
- An executable line of code may not contain the following statements:
  - Whole line comments, including empty comments and banners
  - Blank lines

2. Checklist for source statement counts

PHYSICAL SLOC COUNTING RULES			
MEASUREMENT UNIT	ORDER OF PRECEDENCE	PHYSICAL SLOC	COMMENTS
Executable lines	1	One per line	Defined in 2.8
Non-executable lines			
Declaration (Data) lines	2	One per line	Defined in 2.4
Compiler directives	3	One per line	Defined in 2.5
Comments			Defined in 2.7
On their own lines	4	Not included (NI)	
Embedded	5	NI	
Banners	6	NI	
Empty comments	7	NI	
Blank lines	8	NI	Defined in 2.6

LOGICAL SLOC COUNTING RULES				
NO.	STRUCTURE	ORDER OF PRECEDENCE	LOGICAL SLOC RULES	COMMENTS
R01	"*foreach*", "*while*" or "*if*" statement	1	Count once.	"*while*" is an independent statement.
R03	Statements ending by a semicolon or newline	2	Count once per statement, including empty statement.	

3. Examples

EXECUTABLE LINES

SELECTION Statement

ESS1 – if, else if, else and nested if statements

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

if (<*Boolean expression*>) then
  <*statements*>
endif
	if ($x != 0) then
	  echo “non-zero”
	endif
		1
		1
		0

if (<*Boolean expression*>) then
  <*statements*>
else
  <*statements*>
endif
	if ($x > 0) then
	  echo “positive”
	else
	  echo “negative”
	endif
		1
		1
		0
		1
		0

if (<*Boolean expression*>) then
  <*statements*>
else if (<*Boolean expression*>)
  <*statements*>
else
  <*statements*>
endif
	if ($x == 0) then
	  echo “zero”
	else if ($x > 0) then
	  echo “positive”
	else
	  echo “negative”
	endif
		1
		1
		1
		1
		0
		1
		0

if (<*Boolean expression*>) <*statement*>	if ($x != 0) echo “non-zero”	2
NOTE: complexity is not considered, i.e. multiple “&&” or “||” as part of the expression.		

ESS2 – switch and nested switch statements

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

switch (<*string*>)
  case <*string 1*>:
  <*statements*>
  breaksw
  case <*string 2*>:
  <*statements*>
  breaksw
  case <*string 3*>:
  <*statements*>
  break
  default:
  <*statements*>
endsw
	switch ($str)
	  case “1”:
	  echo “one”
	  breaksw
	  case “2”:
	  echo “two”
	  breaksw
	  case “3”:
	  echo “three”
	  breaksw
	  default:
	  echo “invalid case”
	endsw
		1
		0
		1
		1
		0
		1
		1
		0
		1
		1
		0
		1
		0

ITERATION Statement

EIS1 – foreach

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
foreach <*name*> (<*wordlist*>) <*statements*> End	foreach i ($d) echo $i end	1 1 0

EIS2 – while

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

while <*expression*>
  <*statements*>
end
	while ($j <= 10)
	  echo '2 **' $j = $i
	  @ i *= 2
	  @ j++
	end
		1
		1
		1
		1
		0

JUMP Statement

EJS1 – goto, label

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
goto *label* . label:	loop1: $x++ if ($x < $y) goto loop1	0 1 2

EJS2 – break

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
break	if ($i > 10) break	2

EJS3 – exit function

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
exit *return_code*	if ($x < 0) exit 1	2

EJS4 – continue

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

continue	while ($i < 10)
	  $i++
	  if ($i == 5) then
	  continue
	  else
	  $j++
	  endif
	end
		1
		1
		1
		1
		0
		1
		0
		0

EXPRESSION Statement

EES1 – function call

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
<*function_name*> (<*parameters*>)	read_file (name)	1

EES2 – assignment statement

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
$<name> = <value>	$a = “value”	1

EES3 – empty statement(is counted as it is considered to be a placeholder for something to call attention)

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
one or more “;” in succession	;	1 each