Makefile CodeCount™ Counting Standard

University of Southern California

Center for Systems and Software Engineering

August , 2012

Revision Sheet

Date	Version	Revision Description	Author
6/20/2012	1.0	Original Release	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. There are no explicit data declaration statements in Makefiles.

1.5. **Compiler Directives –** A statement that tells the compiler how to compile a program, but not what to compile.

The following table lists the Makefile keywords that denote compiler directive lines:

include	-include	sinclude

Table 1 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.

Makefile 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:
  - Variable assignment statements
  - Target: prerequisite statements
  - Labeled statements
- An executable line of code may not contain the following statements:
  - Compiler directives
  - 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 1.8
Non-executable Lines			
Declaration (Data) lines	2	One per line	Defined in 1.4
Compiler Directives	3	One per line	Defined in 1.5
Comments			Defined in 1.7
On their own lines	4	Not Included (NI)	
Embedded	5	NI	
Banners	6	NI	
Empty Comments	7	NI	
Blank Lines	8	NI	Defined in 1.6

LOGICAL SLOC COUNTING RULES				
NO.	STRUCTURE	ORDER OF PRECEDENCE	LOGICAL SLOC RULES	COMMENTS
R01	Variable Assignment	1	Count Once	Independent statement
R02	Variable Definition	2	Count once each line	Independent statement
R03	Target	3	Count once each line	Independent statement
R04	Clean	4	Count once each line	Independent statement
R05	Secondary Expansion	5	Count once each line	Independent statement
R06	Compiler Directive	6	Count once per directive	Independent statement

3. Examples

EXECUTABLE LINES
EXPRESSION Statement

EES1 – variable assignment

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
<name> = <value>	OBJECTS = file1.o file2.o file3.o	1
<name> ?= <value>	immediate ?= deferred	1
<name> := <value>	immediate := deferred	1
<name> += <value>	immediate += deferred	1

EES2 – clean statement

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
clean: <statements>	clean: rm -f file1.txt file2.txt	1 1

EES3 – target prerequisite statement

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
target: <prerequisite recipe>	file1.o: file1.c $(CC) -g -c file1.c edit: main.o kbd.o command.o \ display.o insert.o	1 1 1 0

EES4 – secondary expansion

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
.secondexpansion: <statements>	.SECONDEXPANSION: AVAR = top onefile: $(AVAR) twofile: $$(AVAR)	1 1 1 1 1 1

COMPILER DIRECTIVES

CDL1 – directive types

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Include <file_name>	Include makefile1	1