Cascading Style Sheets (CSS) Counting Standard

University of Southern California

Center for Systems and Software Engineering

Fall 2008

Revision Sheet

Date	Version	Revision Description	Author
10/02/08	1.0	Original Release	CSSE

1.0 Checklist for source statement counts

PHYSICAL AND LOGICAL SLOC COUNTING RULES

Measurement Unit	Order of Precedence	Physical SLOC	Logical SLOC	Comments
Executable lines	1	One per line	See table below	Defined in 2.4
Non-executable lines				
Declaration (Data) lines	2	One per line	See table below	Defined in 2.5
Comments				Defined in 2.6
On their own lines	3	Not included (NI)	NI	
Embedded	4	NI	NI	
Empty comments	5	NI	NI	
Blank lines	6	NI	NI	Defined in 2.7

Table 1 Physical and Logical SLOC Counting Counts

LOGICAL SLOC COUNTING RULES

No.	Structure	Order of Precedence	Logical SLOC Rules	Comments
R01	<tagname> { … }	1	Count once.	Each tag is counted once. It can be an HTML tag or a user defined class name.
R02	.. {<keyword> : <value>; … }	2	Count number of semicolon separated statements.	The last statement need not end with a semicolon.

Table 2 Logical SLOC Counting Rules

2. Definitions

2.1 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.

2.2 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.

2.3 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.

2.4 Executable Lines – These include the lines contained within the definition of a CSS class such as *p*, *td* etc or even user defined classes. Every executable line needs to end with a semicolon barring the last line before the container class is closed (using a curly brace)

2.5 Data Declaration Lines – All executable lines are contained within a class. Each class begins and ends with a curly brace. The following example shows how a class can be defined.

```
<tag-name> {
        ......
    }
```

Irrespective of how many executable lines are contained within the curly braces, the number of data declaration lines for the above example is 1. Therefore a data declaration line for CSS corresponds to those lines that define a class.

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

CSS comments start with "/*" and end with "*/". A whole comment line may span one or more lines and does not contain any compilable source code. An embedded comment can co-exist with compilable source code on the same physical line.

2.6 Blank Lines – These are lines that contain only:

- White space characters
- Line feed characters
- Carriage return characters.