

WHITEPAPER FRONTENDART



DIFFERENT DEFINITIONS OF LINES OF CODE

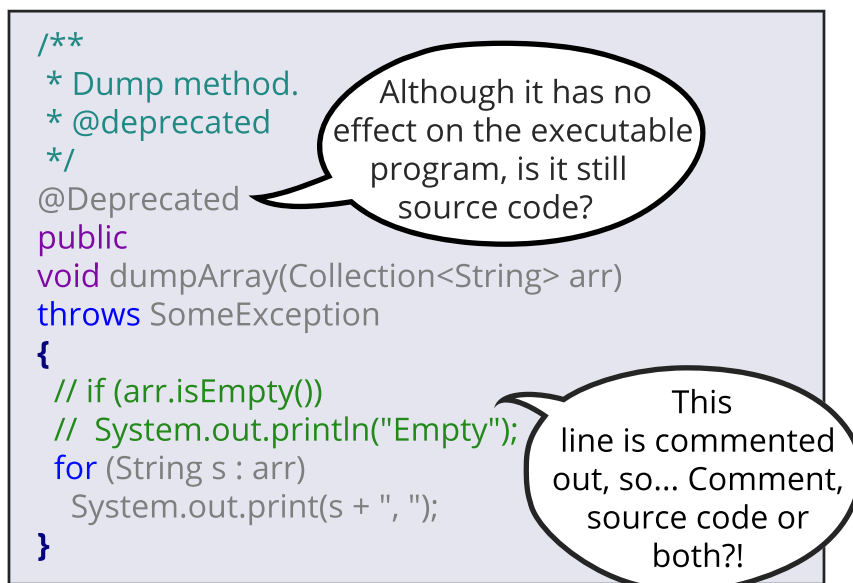
Lines of code

Lines of Code is one of the most controversial source code metrics in software engineering practice. It is relatively easy to calculate, understand and use by the different stakeholders for a variety of purposes – that is why it is the most frequently applied measure in software estimation, quality assurance and many other fields. Yet, there is a high level of variability in the definition and calculation methods of the metric which makes it difficult to use it as a base for important decisions. Furthermore, there are cases when its usage is highly questionable – such as direct programmer productivity assessment.

LOC is generally seen as a measure of system size expressed using the number of lines of its source code as it would appear in a text editor, but the situation is not that simple. However, the importance of this metric is clear: it can be used for different purposes including size estimation, productivity assessment and providing a base for other relative measurements such as defect density.

There is a controversy among professionals what the LOC metric should actually mean and how it should be defined and measured. Here is an incomplete list of the most important definitions from various sources. Do you like syntactic sugar? Would you count it in LOC?

Syntactic sugar make the code easier to read or to express: added features or other formalism to make it "sweeter" for humans, but which do not affect the expressiveness of the formalism.



Definitions from different sources

1. *Wikipedia* – http://en.wikipedia.org/wiki/Source_lines_of_code
 - LOC: "The most common definition of physical SLOC is a count of lines in the text of the program's source code including comment lines. Blank lines are also included unless the lines of code in a section consists of more than 25% blank lines."
 - LLOC: "Logical SLOC attempts to measure the number of executable 'statements.'"

2. *SonarQube tool* – <http://docs.codehaus.org/display/SONAR/Metric+definitions>
 - Lines: “Number of physical lines (number of carriage returns).”
 - Lines of code: “Number of physical lines that contain at least one character which is neither a whitespace or a tabulation or part of a comment.”
3. *Cunningham & Cunningham, Inc.* – <http://c2.com/cgi/wiki?LinesOfCode>
 - LOC: “Lines of Code, usually referring to non-commentary lines, meaning pure whitespace and lines containing only comments are not included in the metric.” (The number of lines of program code is wonderful metric. It can be used as a measure of complexity or productivity.)
4. *NDepend tool* – <http://www.ndepend.com/Metrics.aspx>
 - “NbLinesOfCode: (defined for application, assemblies, namespaces, types, methods) This metric (known as LOC) can be computed only if PDB files are present. NDepend computes this metric directly from the info provided in PDB files. The LOC for a method is equals to the number of sequence point found for this method in the PDB file.”
5. *PC Magazine* – <http://www.pcmag.com/encyclopedia/term/46137/lines-of-code>
 - Lines of Code: “The statements and instructions that a programmer writes when creating a program. One line of this ‘source code’ may generate one machine instruction or several depending on the programming language. A line of code in assembly language is typically turned into one machine instruction. In a high-level language such as C++ or Java, one line of code may generate a series of assembly language instructions, resulting in multiple machine instructions. Lines of Code Are Not the Same: One line of code in any language may call for the inclusion of a subroutine that can be of any size, so while used to measure the overall complexity of a program, the line of code metric is not absolute.”
6. *Eclipse Metrics plugin tool* – <http://metrics.sourceforge.net/>
 - TLOC: “Total lines of code that will counts non-blank and non-comment lines in a compilation unit. Useful for thoses interested in computed KLOC.”

7. *Understand tool* – http://scitools.com/support/metrics_list/

- Lines of Code (Include Inactive): “Number of lines containing source code, including inactive regions.”
- Physical Lines: “Number of physical lines.”
- Source Lines of Code: “Number of lines containing source code. [aka LOC]: The number of lines that contain source code. Note that a line can contain source and a comment and thus count towards multiple metrics.”

There is no common interpretation of the LOC metric and there are even mentionable definitions such as Wikipedia's LLOC, which refers to actual syntactic program elements rather than code lines. Similarly, non-empty lines, declarations and comments are interpreted differently. When using LOC, it is important to state precisely which definition it is based on and to use the same calculation method in case of comparison.