Project #2 – Code Analyzer

Extended from 10/6 to 10/9

Version 2.2

Purpose:

This project develops a program for analysis of C# code. The analyzer, given a set of file references, finds all the types, their members, and relationships with other types. Further, the analyzer constructs two metrics for each member function: its size in lines of code, and complexity defined as the number of elements in its scope tree.

The purpose of this code analysis is to find functions that may need attention because of their size and/or complexity. A secondary purpose is to help developers get a quick understanding of the structure of a set of code.

Requirements:

The Code Analyzer:

1. **(1) Shall** be implemented in C# using the facilities of the .Net Framework Class Library (FCL), version 4.5, as provided in the EECS clusters.

2. **(2) Shall** accept a specification of files using a path, file patterns, and options from the command line. The Analyzer **shall** provide a command line option /S indicating that all subdirectories rooted at the path are to be searched for files matching all supplied patterns. If there is no /S on the command line only the cited path will be searched.

3. **(2) Shall** provide output information that lists types defined within the file set and their function names[1].

4. **(2) Shall** display function sizes, and complexities[2] for each function in each file, and a summary for all the files in the specified set.

5. **(10) Shall** provide a command line option, /R, which, when present causes the analyzer to display the relationships between all types defined in the file set, e.g., inheritance, composition, aggregation, and using relationships instead of the function sizes and complexities.

6. **(2) Shall** display all output on standard output[3]. The analyzer **shall** provide an option, /X, which when present causes the output to also be written to a file in XML format.

7. **(1)** Your submission **shall** supply a compile.bat and run.bat[4] file that compiles and runs your project without the user needing to specify any information, e.g., you are providing default values for path and file set. The program output **shall** demonstrate that you meet requirements #2 - #5.

Please make the path be relative to the path to your project and the files that make up your project. The submission **shall** also supply all the source code, project files, and solution file, for Visual Studio 2013[5].

[1] In this project (Requirements #3 and #4) we do not expect you to detect lambdas, although you may do that if you wish.

[2] Function complexity is the number of scopes the function implements, including braceless scopes.

[3] That will display, by default on the console terminal, but may be redirected to a file.

[4] The run.bat file will need to demonstrate all of the options with successive executions of the project's program.

[5] As a registered student of an EECS program you may download an iso file to install Visual Studio 2013 from Dreamspark.com/Student free of charge.