FUJITSU

FUJITSU Software SIMPLIA MF-STEPCOUNTER JAVA Software Metrics Measurement V60L14

User's Guide

Windows

SIMPLIA-STJ-EN60(07)
November 2018

Preface

SIMPLIA MF-STEPCOUNTER Java Software Metrics Measurement is an application that measures the assets of Java development. This application enables comprehension of the development progress and design quality of classes and methods.

New Features

There are no additional features for this function.

To Read the Help

The latest version of Internet Explorer supported by each operating system is recommended as the web browser for referring to the user's guide.

Registered Trademark Information

The registered trademarks and trademarks used in this user's guide are as follow.

Microsoft, Windows, MS-DOS, and MS are registered trademarks of Microsoft Corporation in the United States and other countries.

- Oracle and Java are registered trademarks of Oracle and/or its affiliates in the United States and other countries. Company names and product names used in this document are registered trademarks or trademarks of those companies.

Abbreviations

The following abbreviations are used in this user's guide.

"Windows(R) 10 Home", "Windows(R) 10 Pro", "Windows(R) 10 Enterprise", or "Windows(R) 10 Education"	-> 	"Windows 10"
"Windows(R) 8.1", "Windows(R) 8.1 Pro", or "Windows(R) 8.1 Enterprise"	-> 	"Windows 8.1"
"Windows(R) 7 Home Premium", "Windows(R) 7 Professional", "Windows(R) 7 Enterprise", or "Windows(R) 7 Ultimate"	-> 	"Windows 7"
"Microsoft(R) Windows Server(R) 2016 Datacenter", "Microsoft(R) Windows Server(R) 2016 Standard", or "Microsoft(R) Windows Server(R) 2016 Essentials"		"Windows Server 2016"
"Microsoft(R) Windows Server(R) 2012 R2 Datacenter", "Microsoft(R) Windows Server(R) 2012 R2 Standard", "Microsoft(R) Windows Server(R) 2012 R2 Essentials", or "Microsoft(R) Windows Server(R) 2012 R2 Foundation"	-> 	"Windows Server 2012 R2"
"Microsoft(R) Windows Server(R) 2012 Datacenter", "Microsoft(R) Windows Server(R) 2012 Standard", "Microsoft(R) Windows Server(R) 2012 Essentials", or "Microsoft(R) Windows Server(R) 2012 Foundation"	-> 	"Windows Server 2012"
"Microsoft(R) Windows Server(R) 2008 R2 Datacenter", "Microsoft(R) Windows Server(R) 2008 R2 Standard", "Microsoft(R) Windows Server(R) 2008 R2 Enterprise", or "Microsoft(R) Windows Server(R) 2008 R2 Foundation"	-> 	"Windows Server 2008 R2"
"Windows 10", "Windows 8.1", "Windows 7", "Windows Server 2016", "Windows Server 2012 R2",	-> 	"Windows"

"Windows Server 2012", or "Windows Server 2008 R2"		
"Java(TM)"	-> 	"Java"
"Java(TM) 2 Platform Standard Edition Development Kit 8.0"	-> 	"JDK" or "JDK8.0"

Contents

Chapter 1 Background and Purpose

In software development, there are many situations where the rate of resolution of bugs is low and degrading occurs due to frequent requirement changes.

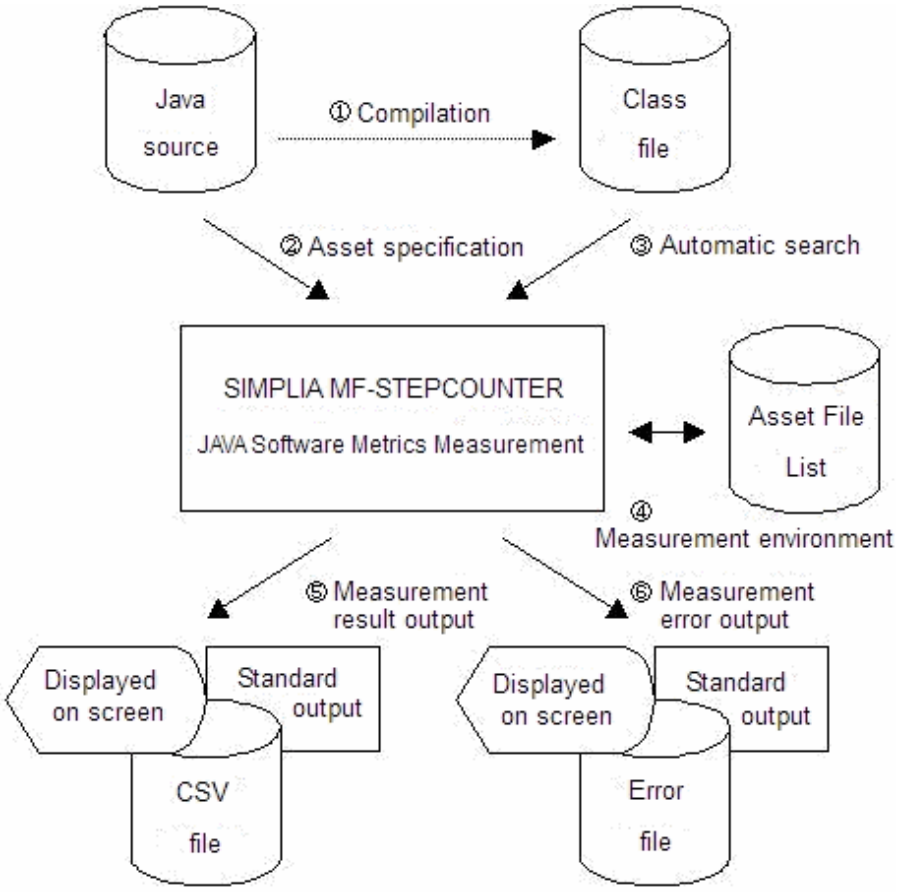
If the cause is source code expansion due to repeated upgrades, it takes a lot of work to understand the processing and it is difficult to perform sufficient testing to verify that no bugs exist when time is limited. It is also difficult to make a design flexible enough to promptly respond to requirement changes.

Quantifying source code quality for quantitative evaluation is important in terms of quality improvement.

Metrics Measurement measures items from Java sources and class files as indicators of class and method quality management and evaluation. This function allows comprehension of the development progress and design quality of classes and methods. It also assists in basic comprehension of the classes and methods to be focused on in testing.

Chapter 2 Overview

This tool performs measurement using the Java sources that are the assets of Java development and the class files that are the results of Java source compilation. Measurement is available by class unit and by method unit, and the measurement results are output for each specified measurement unit. The measurement results are not only displayed on screen but also output in CSV files. The asset file list is used to save the assets measured once to enable measurement in the same environment any number of times. However, it is not possible to measure Java sources based on SQLJ.



1. Compilation

   Java source must be compiled before using this tool.

2. Asset specification

   The Java source that is the target of measurement should be specified by file unit or by folder unit. This tool performs measurement recognizing only files with the .Java extension as the targets of measurement.

3. Automatic search

   For class file measurement, files are identified through Java source analysis and automatically detected using the path name specified in the CLASSPATH environment variable and the class path name specified in the tool, and then measured.

4. Measurement environment:

   The specified environment (output CSV file name/asset names/measurement units/class path names/alarm specification format) can be saved. The asset file list saving the specified environment can be used to perform measurement in the same environment any number of times.

5. Measurement result output

   The measurement results are output for each specified measurement unit.

6. Measurement error output

   Error details are output if a measurement error occurs.

The items output as measurement results are as follow.

(1) When measurement by class unit is specified

No.	Fundamental Measurement	Information Source	Description
1	Class Name	Java Source	The class name that is the measurement unit is output.
2	Number	Java Source	The number of executable statements within the class is measured.
3	Number of Inheritances	Class Files	The number of class hierarchies within the class is measured.
4	Number of Field Variables	Class Files	The number of field variables within the class is measured.
5	Number of Instance Variables	Class Files	The number of instance variables within the class is measured.
6	Number of Public Variables	Class Files	The number of public variables within the class is measured.
7	Number of Methods	Class Files	The number of methods defined within the class is measured.
8	Number of Instance Methods	Class Files	The number of instance methods defined within the class is measured.
9	Number of Public Methods	Class Files	The number of public methods defined within the class is measured.
10	Definition File Name	Java Source	The file name of the Java source defining "1. Class Name" is output. If the Java source contains multiple class definitions, this is only output for the first class in the Java source and not for the other classes.

(2) When measurement by method unit is specified

No.	Fundamental Measurement	Information Source	Description
1	Method Name	Java Source	The method name that is the measurement unit is output.
2	Number	Java Source	The number of executable statements within the method is measured.
3	Number of Branches	Java Source	The number of conditional branches (if statements/case statements within switch) within the method is measured.
4	Number of Loops	Java Source	The number of loops (for statements/while statements/do-while statements) within the method is measured.
5	Number of CALL Methods	Java Source	The number of other method executions is measured.
6	Definition File Name	Java Source	The file name of the Java source defining "1. Method Name" is output. If the Java source contains multiple method definitions, this is only output for the first method in the Java source and not for the other methods.

Chapter 3 Installation Procedure

This chapter explains the procedure for installing SIMPLIA MF-STEPCOUNTER Java Software Metrics Measurement.

3.1 Installation and Uninstallation

For details on the installation and uninstallation of SIMPLIA MF-STEPCOUNTER Java Software Metrics Measurement, refer to the software documentation provided with the product.

3.2 System Requirements

The system requirements are as follows.

Hardware Requirements

Supported Machines
Computers running Windows 10, Windows 8.1, Windows 7, Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, or Windows Server 2008 R2

Software Requirements

Required Software		
Software Name	Applicable Requirements (Version/ Level)	Remarks
OS	Windows 10 Windows 8.1 Windows 7 Windows Server 2016 Windows Server 2012 R2 Windows Server 2012 Windows Server 2008 R2	One of the OSs listed in the left column is required.
JDK	8.0	The PATH environment variable must include the bin folder of JDK.

Chapter 4 Function Description

This chapter provides measurement contents and alarm specification details.

4.1 Description of Measurement Contents

This section explains the contents of measurement by class unit and measurement by method unit.

4.1.1 Measurement by Class Unit

In measurement by class unit, the Java file included in the specified asset is measured by class unit. After Java file measurement, class files are measured from the class names defined in the Java file. The class files are automatically searched from the search path specified in the CLASSPATH environment variable and the search path specified as the class path in this tool and then measured.

Note: The folder structure in a situation where class files are expanded to folders and the folder structure within a JAR file must be consistent with that of the package as with classpath specification in the normal Java Runtime Environment.

If the measurement target class uses/inherits a class included in the user's own or the system's class library (such as servlet.jar), the name of the folder/Jar file containing the package must also be set as the search path for the CLASSPATH environment variable or the class path within this tool.

In addition, normal measurement is achieved if the class files and the Java file have been synchronized and kept up-to-date. Before performing measurement by class unit, confirm that the Java file has already been compiled.

Description of Output Items

Output Item	Description
Class Name	The class name as a measurement unit. It is output in the format of "Package Name.Class Name." The class names are output in the order of their definition in the Java file. If the Java file contains internal class and anonymous class definitions, the class names are generated according to the following rules. Rule 1 For an internal class defined directly in the class, Source Class$Internal Class is used to generate the class name. Rule 2 For an internal class defined in a method, Source Class$Serial Number$Internal Class is used to generate the class name. Rule 3 For an anonymous class, Source Class$Serial Number is used to generate the class name. * The serial numbers are to refer to the order of appearance and avoid class name overlapping.
Number	The number of statements within the class.
Number of Inheritances	The number of inheritance classes.
Number of Field Variables	The number of field variables defined within the class.
Number of Instance Variables	The number of instance variables defined within the class.
Number of Public Variables	The number of public variables defined within the class.
Number of Methods	The number of methods defined within the class.
Number of Instance Methods	The number of instance methods defined within the class.
Number of Public Methods	The number of public methods defined within the class.
Definition File Name	The name of the Java file defining the class. This is output only for the class name defined first in the Java file.

Measurement Criteria

Measurement Item	Description	Measured File
Number	The number of executable statements. This is not the number of lines. In addition, comment lines/package statements/import statements/class definition statements/method definition statements/field variable definition statements are not included.	Java
Number of Inheritances	The number of hierarchies inheriting higher classes.	Class
Number of Field Variables	The number of all instance and static variables.	Class
Number of Instance Variables	The number of variables with no static modifier.	Class
Number of Public Variables	The number of variables with the public modifier.	Class
Number of Methods	The number of all methods.	Class
Number of Instance Methods	The number of methods with no static modifier.	Class
Number of Public Methods	The number of methods with the public modifier.	Class

Two-byte hyphens (-) are output for the measurement items if class file measurement fails.

Examples of judgment with these measurement criteria are shown below.

```
Java File Description                      Judgment
package TEST.Package;
class TEST01 extends TEST00 {          -> Number of Inheritances is 2 if TEST00 is the top level class
  int nDec;                             -> Instance Variable/Field Variable
  static public String str = "TEST01";  -> Public Variable/Field Variable
  public TEST01() {
    nDec = 0;                           -> Number
  }
  int getDec() {                        -> Number of Instance Methods/Number of Methods
    return nDec;                        -> Number
  }
  static public void printStr() {       -> Number of Public Methods/Number of Methods
    System.out.println(str);            -> Number
  }
  private String toDecString(){         -> Number of Instance Methods/Number of Methods
    return Integer(nDec).toString();    -> Number
  }
};
```

4.1.2 Measurement by Method Unit

In measurement by method unit, Java files included in the specified asset are measured by method unit. Class file measurement is not included in this measurement.

Description of Output Items

Output Item	Description
Method Name	The method name as a measurement unit. It is output in the format of "Package Name.Class Name.Method Name (Method Argument)."
Number	The number of statements within the method. This measurement item does not refer to the number of lines.
Number of Branches	The number of conditional branches within the method. This is the number of if statements (else if statements)/case statements within a switch statement, or the number of And/Or

Output Item	Description
	conditions of these if the -t option is specified for the start option. Single else statements and the default statement within a switch statement are excluded.
Number of Loops	Number of Loops This is the number of for statements/while statements/do while statements, or the number of And/Or conditions of those statements.
Number of CALL Methods	The number of other method executions.
Definition File Name	The name of the measurement target Java file defining the method. This is output only for the method name defined first in the Java file.

Measurement Criteria

Measurement Item	Description	Measured File
Number	The number of executable statements. This is not the number of lines. In addition, comment lines are not included.	Java
Number of Branches	This is the number of if statements (else if statements)/case statements within a switch statement, or the number of And/Or conditions in the if statements if the -t option is specified for the start option. Single else statements and the default statement within a switch statement are excluded.	Java
Number of Loops	This is the number of for statements/while statements/do while statements, or the number of And/Or conditions of those statements.	Java
Number of CALL Methods	The number of other method executions.	Java

Examples of judgment with these measurement criteria are shown below.

```
Java File Description                    Judgment
package TEST.Package;
class TEST01 extends TEST00 {
  int nDec;
  static public String str = "TEST01";

  static public void printStr() {
    String strOut = "";                  -> Number
    for(int i=0;i<10;i++) {              -> Number of Loops/Number
      if( i==0 && nDec > 0 ) {          -> Number of Branches (2 if And/Or conditions are measured;
1 if not)/Number
        strOut += "CLASS";               -> Number
      } else {
        strOut += str;                   -> Number
      }
    }
    print(strOut);                       -> Number of CALL Methods/Number
  }
  static private void print(String str) {
    System.out.println(str);             -> Number
  }
};
```

4.2 Alarm Specification

This section provides descriptions of judgment on measurements, criteria configuration examples, and representation of percentiles.

4.2.1 Judgment on Measurements

Overview

The contents of measurement with this tool enables comprehension of the progress of development from the target level and the actual development quantity, based on the number of statements in each class or method. Furthermore, the measurement contents can also be used to judge the design quality of classes and methods. Judgment details are shown below.

Judgment Details

1. If the Number of Methods Is Too Large

   If the number of methods is too large, the class may contain too many functions. If the number of methods is large on the whole, it may be difficult to understand the contents and be hard to maintain or reuse the methods. In this case, it is necessary to review the concept of class units and examine whether the classes can be subdivided into appropriate units. If a certain class or class group (package) is enormous and has caused the average value to increase, it may be necessary to review function sharing with other parts.

2. If the Number of Variables Is Too Large

   Basically, there are the same issues as with the number of methods. However, for constant fields, it is not necessary to make an issue of comparatively large numbers. Also, Dialog and DB entity classes are much larger than the others in size.

3. If Number Is Too Large

   As mentioned above, the class may contain too many functions. In particular, if the number is large in comparison with the number of methods, it is assumed that the program is likely to be procedure-centric. In this case, it is necessary to reconsider whether the methods have been divided properly as functions with objects. If only a specific large process is made a method, it is hard to use it as a part and reusability is lowered. If this pattern is seen overall, it is necessary to reconsider whether the methods have been divided properly from the viewpoint of large level design/modeling.

4. If the Number of Methods, the Number of Field Variables, and Number Are Too Small

   If these are too small, the class is less valuable as a part and less reusable. In addition, overhead by division becomes larger, and it is not preferable in terms of runtime performance.

5. If the Number of Inheritances Is Too Deep

   If too many inheritances are used, readability of the program is lowered and it is difficult to perform maintenance. It is necessary to review the design to make the depth appropriate. If too many inheritances are used, it may be difficult to change higher level classes, and it is necessary to consider the division of functions by transfer or other means. However, when a system class of a deep inheritance hierarchy is inherited and used, user-defined hierarchies are targeted for judgment.

6. If Inheritances Are Rarely Used

   If inheritances are rarely used, the developer may have a poor grasp of the object-oriented concept. In combination with class size information, functions must be examined to determine whether there are any parts that can be converted into subclasses.

4.2.2 Alarm Criteria Configuration Examples

Overview

This section provides examples of configuration of alarm criteria values in the measurement results dialog below, based on business software case examples.

Criteria Values

To examine criteria values for Java program alarm specification, business software case examples such as the following have been measured.

- **A** Software Development and Order Management System (Company A)

  1132 files, about 318K lines, 1177 classes

- **B** Bill of Material System (Company B, Client Only)

  709 files, about 59K lines, 799 classes
  This is divided into phase 1 and phase 2.

- **C** Sales Force Automation (Company C)

  132 files, about 36K lines, 186 classes

- **D** System (Company D, Client Only)

  44 files, about 8K lines, 44 classes

- **E** Design Work for Manufacturers Inter-department Workflow (Company E)

  136 files, about 12K lines, 161 classes

The results were categorized by dialog (panel) class probably representing a big feature in the case of office automation software, by entity class (equivalent to a DB table), and by other classes and then measured. Categorization was based on the package and class names. In general, standard deviation is used to view features of distribution, but it is not valid for this tool because usually the data obtained through measurement with this tool is not normally distributed. Here, the nature of distribution was examined based on quartile deviation and then reference average values and upper limits were set. Java criteria values are as shown in the table below.

Table 4.1 Java Criteria Values

Item	Dialog (Average, Upper Limit)	DB Entities (Average, Upper Limit)	General (Average, Upper Limit)
Number of Fields	15 - 50,100	20,80	3 - 10,20
Number of Methods	10 - 35,80	40,160	5 - 10,30
Number of Statements	100 - 400,700	100,300	25 - 80,300
Number of Non-private Non-constant Fields	0	0	0

- Average

  This is a reference average value in a certain unit and used for the investigation of features of the entire program or a package. If a value deviates from this value significantly, the corresponding part may have a problem in its design.

- Upper Limit

  This gives an indication to help determine whether the class or function is abnormal. Classes and functions exceeding this value probably have problems in their structure, so it is better to hold reviews. These criteria have only been set as rough indications and are not absolute criteria. For practical application, it is desirable to configure criteria settings according to the needs of individual projects. For example, criteria are obviously different between when a normal application is designed and when a versatile library is designed with a strong awareness of reusability. The criteria above are intended for normal applications. As for reusable libraries, the number of methods, etc. tend to be larger.

Features by category are as follows.

The total size of dialog classes tends to be larger. Although there are various restrictions such as requirements for dialog design and features of codes generated by the GUI builder in use, it is a good idea to use the criteria levels above.

As a DB entity class is based on the DB table size, the number of fields and the number of methods are often large. Generally, data access methods will be central, so the number of statements can be relatively low. Regarding this, the criteria are provisional values, and there are many obscure points.

The criteria for general Java classes are considered to be the criteria values for other classes.

For your reference, C++ criteria values are shown below.

Table 4.2 C++ Criteria Values

Item	Reference Average	Upper Limit
Number of Data Members (= Fields)	2 - 4	10
Number of Method Functions (= Methods)	3 - 13	30
Number of Statements	20 - 120	500

The aggregate results of the number of fields/the number of methods/the number by classification are shown below.

1. Dialog

   For dialog classes, the results of relatively larger A and B have been adopted to set criteria values.

Table 4.3 A (131 Classes, 1 Interface)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	6503	49.27	253	0	59.90	29.0	57	235
Number of Methods	3413	25.86	120	2	21.45	19.5	23	104
Number	62749	475.37	2897	0	521.22	307.0	609	2534

Table 4.4 B (Phase 1) (50 Classes, 0 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	715	14.30	46	0	14.90	11.5	26	104
Number of Methods	524	10.48	24	1	6.69	11.0	12	52
Number of Statements	4958	99.16	309	3	89.21	78.5	146	600

Table 4.5 B (Phase 2) (7 Classes, 0 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	197	28.14	48	14	11.95	29.0	22	103
Number of Methods	157	22.43	33	16	7.61	17.0	15	76
Number of Statements	2034	290.57	463	162	97.00	282.0	119	710

2. DB Entity

   For DB entities, large variations were observed depending on the methods involved. For database entities, criteria values have been set based on the values of the class named XXXEntityImpl of A.

Table 4.6 A (275 Classes)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	5657	20.57	164	1	26.86	10.0	18	78
Number of Methods	12695	46.16	334	4	53.58	24.0	35	157
Number of Statements	24723	89.90	682	5	110.94	48.0	64	286

3. Others

For other general classes, in addition to these case examples, criteria have been set by reference to the measurements of the Java standard library, the measurements of two other types of middleware software, and C++ related predetermined values.

Table 4.7 A (238 Classes, 11 Interface)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	2169	7.66	254	0	21.10	3.0	6	25
Number of Methods	2543	8.99	78	0	9.78	7.0	7	31
Number of Statements	17558	62.04	996	0	103.80	33.0	39	173

Table 4.8 B (Phase 1) (646 Classes, 23 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	4301	6.66	145	0	11.05	3.0	7	29
Number of Methods	6503	10.07	138	0	11.82	7.0	10	43
Number of Statements	50359	77.96	1319	0	143.67	33.5	72	300

Table 4.9 B (Phase 2) (96 Classes, 0 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	699	7.28	99	0	13.54	2.0	8	33
Number of Methods	843	8.78	116	0	16.00	5.0	9	37
Number of Statements	7863	81.91	1496	0	181.70	31.0	84	340

Table 4.10 C (74 Classes, 0 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	243	3.28	66	0	9.12	1.0	0	1
Number of Methods	507	6.85	137	0	18.60	3.0	2	10
Number of Statements	1958	26.46	469	0	79.37	3.0	2	11

Table 4.11 D (30 Classes, 3 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	295	9.83	64	0	14.24	4.0	12	48

Number of Methods	190	6.33	31	0	7.20	4.0	7	29
Number of Statements	2402	80.07	398	0	97.57	49.0	123	500

Table 4.12 E (125 Classes, 8 Interfaces)

Measurement Item	Total	Average	Maximum	Minimum	Standard Deviation	Median	Quartile Deviation	Outlier
Number of Fields	802	6.42	53	0	10.15	3.0	6	25
Number of Methods	562	4.50	18	0	3.63	4.0	4	18
Number of Statements	3352	26.82	145	0	31.43	18.0	35	144

4.2.3 Percentile Alarm Indication

Overview

For the measurements displayed in the measurement results dialog, percentile alarm indication is performed.

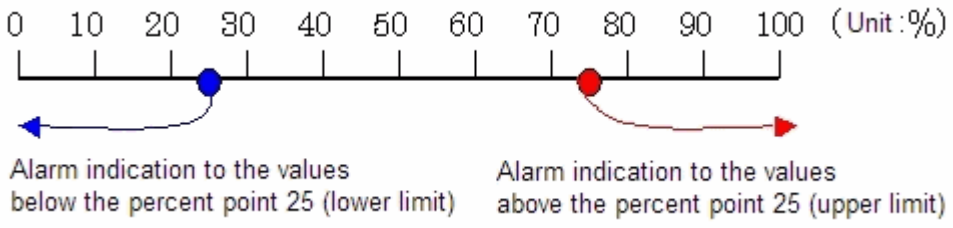
Description of Display

A percentile is a statistical representation to assess values at given percentages after sorting measured data distributions according to their size.

In percentile alarm indication, this function calculates the p percentile (lower limit)/the 100 - p percentile (upper limit) of the measurement class based on the lower percent point specified in the Specify Alarm dialog box and applies alarm indication to the values below the lower limit or above the upper limit.

An example of the range of percentile alarm indication is given below.

Alarm indication range example: 25th percentile



An example of percentile alarm indication is given below.

Example: Assume that there are 10 classes and the number of each is as follows. Percentiles (upper limit/lower limit) in the case where the percent point 30 (lower limit) is specified are obtained, and alarm indication is performed.

[Number of each class] 15,63,84,29,43,55,90,3,72,38

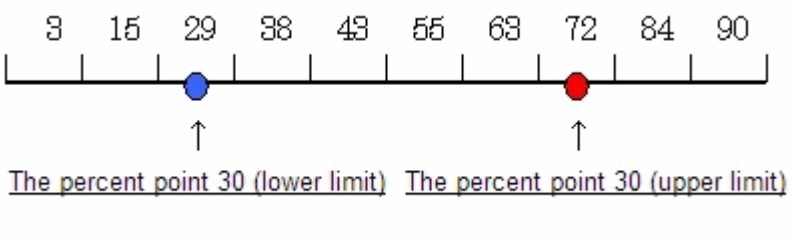
- Sorting of the above data based on size

  3,15,29,38,43,55,63,72,84,90

- Calculation of percentiles (upper limit/lower limit)

  The lower limit is the 3rd one among 10 classes ... Number = 29

  The upper limit is the 8th one among the 10 classes ... Number = 72



- Alarm indication

  Alarm indication is given if the number is below 29 (lower limit).

  Alarm indication is given if the number is above 72 (upper limit).

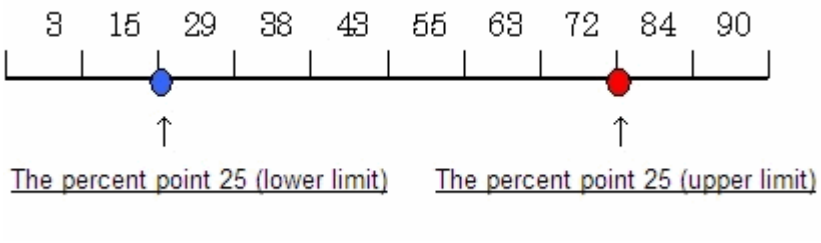
[Number of each class] 15,63,84,29,43,55,90,3,72,38

Example: In the above example, percentiles (upper limit/lower limit) in the case where the percent point 25 (lower limit) is specified are obtained, and alarm indication is performed.

- Calculation of percentiles (upper limit/lower limit)

  The lower limit is between the 2nd one (15) and the 3rd one (29) among 10 classes ... Number = 22

  The upper limit is between the 8th one (72) and the 9th one (84) among 10 classes ... Number = 78



- Alarm indication

  Alarm indication is given if the number is below 22 (lower limit).

  Alarm indication is given if the number is above 78 (upper limit).

[Number of each class] 15,63,84,29,43,55,90,3,72,38

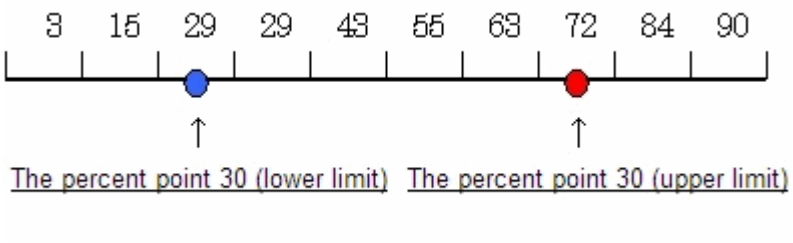
Example: The percent point 30 (lower limit) in the case where overlapping data exists is obtained, and alarm indication is performed.

[Number of each class] 3,15,29,29,43,55,63,72,84,90

- Calculation of rates on the assumption that the overlapping 29s above are both counted and the total count is 10
- Calculation of percentiles (upper limit/lower limit)

  The lower limit is the 3rd one among 10 classes ... Number = 29

  The upper limit is the 8th one among 10 classes ... Number = 72



- Alarm indication

  Alarm indication is given if the number is below 29 (lower limit).

  Alarm indication is given if the number is above 72 (upper limit).

[Number of each class] 3,15,29,29,43,55,63,72,84,90

Chapter 5 Operation Description

This chapter explains how to operate SIMPLIA MF-STEPCOUNTER Java Software Metrics Measurement.

5.1 Starting and Stopping

Start and stop this tool using the following procedures.

How to Start

1. Add the mfstpjm.jar (JDK 8.0) file located in the installation folder of this tool to the CLASSPATH environment variable.

```
Addition example (JDK 8.0):
set CLASSPATH=[Installation Folder]\mfstpjm.jar;%CLASSPATH%
```

2. Set the installation folder of this tool as the current folder, and run the startup class of this tool using java.exe. The startup class of this tool is named com.fujitsu.simplia.StepCounter.metrics. Specify parameters required for the startup if there are any. For details on parameters, refer to "5.4 Command Line".

```
Startup example:
cd [Installation Folder]
java com.fujitsu.simplia.StepCounter.metrics [Parameters]
```

Note: Before executing this tool, the PATH environment variable must be configured to include the bin folder of JDK.

How to Stop

To stop this tool, select [File] on the menu bar and select [Exit SIMPLIA MF-STEPCOUNTER] from the pull-down menu.

5.2 Screen Processing

This section explains how to perform various measurements in screen processing.

5.2.1 New Asset File List Specification Method

Overview

Configure various settings for the new measurement target.

Operating Procedure

1. In the Main dialog, select the [New] command on the [File] menu.
2. Configure various settings (*).
3. In the Main dialog, select the [Save As] command on the [File] menu and save the asset file list.

* The settings are as follows.

Settings Dialog	Setting Item	Description
Main Dialog	CSV File	Specify the measurement results output file name.
	Asset Type	Select the type of the target.
	Asset Name	Specify the measurement target file or folder.
Specify Class Path Dialog	Class Path	Specify the search path for class files. (For measurement by class unit, specify the path to the class file in the CLASSPATH environment variable. For measurement by method unit, this can be left unset.)

Settings Dialog	Setting Item	Description
Specify Measurement Options Dialog	Measurement Unit	Select the measurement unit. (The default setting is class unit.)
Specify Alarm Dialog	Alarm Display Format	Set the display format of the alarm given when the measurement results are displayed. (By default, Percentile is selected and Threshold is set to 25%.)

5.2.2 Target Asset Specification Method

Overview

Specify the measurement target asset names.

Operating Procedure

1. In the Main dialog, select the specification format of the measurement target asset from the [Asset Type] combo box.
2. In the Main dialog, click the [Browse] button next to [Asset Name], and the Select File dialog box is displayed. In the Select File dialog box, select the asset name. The asset selected is of one of the following types, depending on the asset type specified in 1.

[Asset Type] Selection	Selection from [Browse] next to [Asset Name]
File Type	Select the measurement target Java file name. In the Select File dialog box, select the target file directly.
Folder Type (Excluding subfolders)	Select the measurement target folder name. In the Select Folder dialog box, select the target folder directly.
Folder Type (Including subfolders)	Same as above

3. In the Main dialog, click the [Add] button to add the asset name selected in 2 to the [Asset List]. Assets not added to the [Asset List] are exempt from measurement.

5.2.3 Asset File List Reading Method

Overview

Read an asset file list created previously.

Operating Procedure

1. In the Main dialog, select the [Open Asset File List] command on the [File] menu.
2. From the displayed Select File dialog box, select the desired asset file list.

* If an asset file list created in the old version of the tool (V50L10) is selected, alarm indication is given in the [Upper Limit / Lower Limit] format.

5.2.4 Class Path Specification Method

Overview

This tool identifies the class names from the Java file specified as the asset in measurement by class unit and then searches for and measures class files. Therefore, specify the search path (folder name/JAR file name) for class files as the class path.

If the CLASSPATH environment variable is present, the paths specified in CLASSPATH are also searched, so it is not necessary to specify the contents of the CLASSPATH environment variable as the class path. If the CLASSPATH environment variable includes all necessary search paths or if measurement by method unit is to be performed, it is also not necessary to specify this parameter.

Class files are searched according to the order of specification in the CLASSPATH environment variable and the order of specification in the Specify Class Path.

Note: The folder structure in a situation where class files are expanded to folders and the folder structure within a JAR file must be consistent with that of the package as with classpath specification in the normal Java Runtime Environment.

Note: If the length of the full path to a class file or JAR file exceeds the upper limit on the OS, an error may occur due to inability to access the file.

Operating Procedure

1. In the Main dialog, select the [Specify Class Path] command from the [Options] menu.
2. In the Specify Class Path dialog box, enter the folder name or the JAR file name to be set as the class path. Alternatively, click the [Browse Files] button or the [Browse Folders] and make a selection from the Select File dialog or the Select Folder dialog.

   a. Select File dialog

   Select the Jar file name to be set as the class path. The selected JAR file name is set in the [Class Path] text field.

   b. Select Folder dialog

   Select the folder name to be set as the class path. The selected folder name is set in the [Class Path] text field.

   Example 1: When the class name is aaa.class and the package is com.xxx.yyy, and the folder structure is as follows

   ```
   c:\classes\com\xxx\yyy\aaa.class -> Specify c:\classes in folder selection
   ```

   Example 2: When aaa.class is compressed in a JAR file and located as follows

   ```
   c:\classes\jar\abc.jar -> Specify c:\classes\jar\abc.jar in file selection
   ```

   Example 3: When aaa.class inherits the com.xxx.bbb class

   ```
   c:\classlib\com\xxx\bbb.class -> Also specify c:\classlib in folder selection
   ```

3. In the Specify Class Path dialog box, click the [Add] button to add the class path selected in 2 to the [Class Path List]. If it is added to [Class Path List], the class path is set as a search path.
4. After setting all necessary class paths, in the Specify Class Path dialog, click the [OK] button to save the settings and close the dialog box.

5.2.5 Measurement Unit Specification Method

Overview

Specify the measurement unit for the specified assets.

Operating Procedure

1. In the Main dialog, select the [Specify Measurement Option] command from the [Options] menu.
2. From the displayed Specify Options dialog box, select the measurement unit (by class/by method). The [Measurement And/Or Condition] option is also selectable.
3. In the Specify Options dialog box, click the [OK] button to save the settings and close the dialog box.

5.2.6 Alarm Indication Specification Method

Overview

Specify alarm indication for measurements displayed in the measurement results dialog.

Operating Procedure

1. In the Main dialog, select the [Specify Alarm] command from the [Options] menu, or in the measurement results dialog, select the [Specify Alarm] command from the [Options] menu.
2. Using the [Percentile] radio button or the [Upper Limit / Lower Limit] radio button, select the alarm display format.
3. Set target items for alarm indication.

   a. If Percentile is selected

   All measurement items are the targets of alarm indication. No setting of target items is needed.

   b. If Upper Limit / Lower Limit is selected

   From the [Item] combo box in the Specify Alarm dialog box, select measurement items to be the targets for alarm specification.
4. Set the alarm range.

   a. If Percentile is selected

   From the [Threshold] text field in the Specify Alarm dialog box, set the alarm range (lower threshold*). This function calculates the percent point (upper and lower limits) per item based on the specified threshold, and applies alarm indication to the measurements above the upper limit and below the lower limit in the Measurement Results dialog. For details, refer to "4.2.3 Percentile Alarm Indication".

   b. If Upper Limit / Lower Limit is selected

   Set the alarm range using the [Upper Limit] and the [Lower Limit] text fields in the Specify Alarm dialog box. This function applies alarm indication to the measurements above the upper limit and below the lower limit in the measurement results dialog.
5. In the Specify Alarm dialog box, click the [OK] button to save the settings and close the dialog box.

* The lower threshold should be set to a value in the range of 0% - 50%.

5.2.7 Measurement Results Display Method

Overview

View the results of asset measurement.

Operating Procedure

Measurement and measurement results display

1. In the Main dialog, select the [Measure] command from the [Measure] menu to measure the assets.
2. After the measurement is complete, the Measurement Results dialog is opened and the results are displayed.

Display of measurement results

In the Main dialog, select the [Display Results] command from the [Display] menu to open the Measurement Results dialog.

* Note that this operation is valid after the measurement process in the current session of this tool. The measurement results from the previous session of this tool cannot be displayed.

Note also that only the results from the last measurement process can be displayed.

5.2.8 Asset File List Saving Method

Overview

Save the asset file list.

Operating Procedure

New creation and file name changes

1. In the Main dialog, select the [Save As] command from the [File] menu.

2. In the displayed Select File dialog box, specify the destination file name, and click the [OK] button.

Saving

In the Main dialog, select the [Save] command from the [File] menu.

5.2.9 Measurement Method

Overview

Measure the specified assets.

Operating Procedure

In the Main dialog, select the [Measure] command from the [Measure] menu.

The measurement results are output to the CSV file specified in the Main dialog before the measurement.

5.3 Batch Processing

This section explains how to perform various measurements in batch mode.

5.3.1 File Specification Based Measurement Method

Overview

Through command line argument specification, perform the measurement of the specified Java file/class files in batch mode.

Operating Procedure

Specify command line arguments as follows and start this tool.

```
-b [Java File Name]
```

Execution example:

When c:\Java\test.java is the target Java file

```
java com.fujitsu.simplia.StepCounter.metrics -b c:\Java\test.java
```

* The measurement results are output to standard output.

5.3.2 Folder Specification Based Measurement Method

Overview

Through command line argument specification, perform the measurement of Java files contained in the specified folder and class files defined in the Java files in batch mode.

Operating Procedure

Specify command line arguments as follows and start this tool.

```
-b -d [Folder Name]
```

Execution example:

When c:\Java is the target folder

```
java com.fujitsu.simplia.StepCounter.metrics -b -d c:\Java
```

* Specify -s as part of the arguments to also measure subfolders contained in the specified folder.

5.3.3 Asset File List Based Measurement Method

Overview

Through command line argument specification, perform the measurement of Java files specified in the asset file list and class files defined in the Java files in batch mode.

Operating Procedure

Specify command line arguments as follows and start this tool.

```
-b -l [Asset File List Name]
```

Execution example:

If c:\stj\test.stj is the target asset file list

```
java com.fujitsu.simplia.StepCounter.metrics -b -l c:\stj\test.stj
```

* The asset file list is a file storing the asset names, class paths, and options specified in screen processing.

5.3.4 Measurement Unit Specification Method

Overview

Specify the measurement unit through command line argument specification and perform measurement in batch mode.

Operating Procedure

1. To perform measurement by class unit, specify command line arguments as follows and start this tool. By default, measurement by class unit is performed.

   ```
   -b [Asset Parameter]
   ```

   Execution example:

   When c:\Java\test.java is the target Java file

   ```
   java com.fujitsu.simplia.StepCounter.metrics -b c:\Java\test.java
   ```

2. To perform measurement by method unit, specify command line arguments as follows and start this tool.

   ```
   -b [Asset Parameter] -m
   ```

   Execution example:

   When c:\Java\test.java is the target Java file

   ```
   java com.fujitsu.simplia.StepCounter.metrics -b c:\Java\test.java -m
   ```

* The asset parameter refers to file specification/folder specification. If the asset file list is specified for the asset parameter, the measurement unit cannot be specified. In this case, the measurement unit specified in the asset file list is enabled.

5.3.5 Measurement Results Output Specification Method

Overview

Through command line argument specification, specify the measurement results output destination and perform measurement in batch mode.

Operating Procedure

Specify command line arguments as follows and start this tool.

```
-b [Asset Parameter] -csv [CSV File Name]
```

Execution example:

When c:\Java\test.java is the target Java file and c:\csv\test.csv is the destination CSV file

```
java com.fujitsu.simplia.StepCounter.metrics -b c:\Java\test.java -csv c:\csv\test.csv
```

* The asset parameter refers to file specification/folder specification. If the asset file list is specified for the asset parameter, the output destination cannot be specified. In this case, the output destination specified in the asset file list is enabled.

* The CSV file name should be the name of a file that can be written to.

* If no -csv parameter is specified, the measurement results are output to standard output.

5.4 Command Line

This section explains the various command line arguments.

5.4.1 Command Line Arguments

Command line arguments include the following parameters. The order in which parameters are specified is arbitrary.

- Startup Parameters
- Asset Parameters
- Class Path Parameters
- Optional Parameters
- CSV Parameter
- ERR Parameter

As some parameters are exclusive, caution is necessary in specification.

The symbols used for describing the specification method of each parameter are as follow.

Symbol	Description
[]	This parameter is optional.
{}	One of several parameters should be selected and specified.
Underline	This is the default value. If the parameter is omitted, the default value is assumed.

5.4.1.1 Startup Parameters

Specification Method

```
            -b
[{                      }]
            -f
```

Description

Specify how to start this tool. The default value is -f.

Specification Details	Description
-b	Starts batch processing. This parameter requires specification of the asset parameter.
-f	Starts screen processing. This parameter has an exclusive relationship with the asset/class path/optional/CSV parameters.

5.4.1.2 Asset Parameters

Specification Method

```
            -l Asset File List Name
[{          -d Folder Name                }]
            File Name
```

Description

Specify the measurement target assets.

Specification Details	Description
-l (l as in log) Asset File List Name	Targets the contents stored in the asset file list for measurement. The asset file list is the file that stores assets/class paths/CSV output destinations/measurement options. When this parameter is specified, class path/optional/CSV parameters cannot be specified.
-d Folder Name	Targets all files located in the specified folder for measurement. (To include subfolders in the measurement, specify -s as an optional parameter.)
File Name	Specifies the file name of the measurement target Java file. This parameter supports wildcard specification (file specification using * or ?). If a wildcard is specified, the corresponding Java files are measured.

5.4.1.3 Class Path Parameters

Specification Method

```
          Folder Name            Folder Name
[  -c {                } [ ; {                } ] ... ]
          jar File Name          jar File Name
```

Description

This tool identifies the class names from the Java file specified as the asset in measurement by class unit and detects and measures class files. Specify the search path (folder name or JAR file name) for class file detection.

Specification Details	Description
Folder Name	Specifies the folder name to be the search path for class files.
JAR File Name	Specifies the file name of the JAR file that stores class files.

To specify multiple search paths, separate the paths with a semicolon (;) for continuous specification. If the CLASSPATH environment variable is present, the paths specified in CLASSPATH are also searched, so it is not necessary to specify the contents of the CLASSPATH environment variable with this parameter. If the CLASSPATH environment variable includes all necessary search paths or if measurement by method unit is to be performed, it is also not necessary to specify this parameter.

Class files are searched according to the order of specification in the CLASSPATH environment variable and the order of specification with this parameter.

5.4.1.4 Optional Parameters

Specification Method

```
[ -m [ -t ] ] [ -s ]
```

Description

Specify measurement options.

Specification Details	Description
-m	Specifies to perform measurement by method unit. By default, measurement by class unit is performed. When this parameter is specified, class path parameter specification is disabled.
-t	Specifies to measure the number of branches at each And/Or condition in a statement such as an if statement. By default, multiple conditions are recognized as one condition in measurement. This parameter requires -m parameter specification.
-s	This parameter is enabled if the folder name is specified for the asset parameter. If this parameter is specified, JAR files contained in subfolders of the asset (folder) are also measured. If the file name is specified for the asset parameter, specification of this parameter is disabled.

5.4.1.5 CSV Parameter

Specification Method

```
[ -csv CSV File Name ]
```

Description

Specify the output destination for the measurement results.

Specification Details	Description
CSV File Name	Specifies the CSV file name to be the output destination of the measurement results. By default, the measurement results are output to standard output.

5.4.1.6 ERR Parameter

Specification Method

```
[ -efile ]
```

Description

Specify the output destination for measurement errors.

Specification Details	Description
-efile	If this parameter is specified, errors that occurred during measurement are output to an error file. The error file is named "mfstpjm.err" and is output to the temp folder in the installation folder of this tool. By default, errors that occurred during measurement are output to standard output. In addition, even if this parameter is specified, when an error is detected in the error file, errors that occurred during measurement are output to standard output.

5.4.2 Parameter Exclusive Relationships

The exclusive relationships between parameters are shown below.

Parameter		ERR	CSV	Optional		Class Path	Asset			Startup	
		-efile	-csv ~	-s	-m	-c ~	File Name	-d ~	-l ~	-f	-b
Startup	-b	P	P	P	P	P	R (Select one)			N	N

Parameter		ERR	CSV	Optional		Class Path	Asset			Startup	
		-efile	-csv ~	-s	-m	-c ~	File Name	-d ~	-l ~	-f	-b
	-f	P	N	N	N	N	N	N	N	N	N
Asset	-l ~	P	N	N	N	N	N	N	N	-	-
	-d ~	P	P	P	P	P	N	N	-	-	-
	File Name	P	P	D	P	P	N	-	-	-	-
Class Path	-c ~	P	P	P	P	N	-	-	-	-	-
Optional	-m	P	P	P	N	D	-	-	-	-	-
	-s	P	P	N	-	-	-	-	-	-	-
CSV	-csv ~	P	N	-	-	-	-	-	-	-	-
ERR	-efile	N	-	-	-	-	-	-	-	-	-

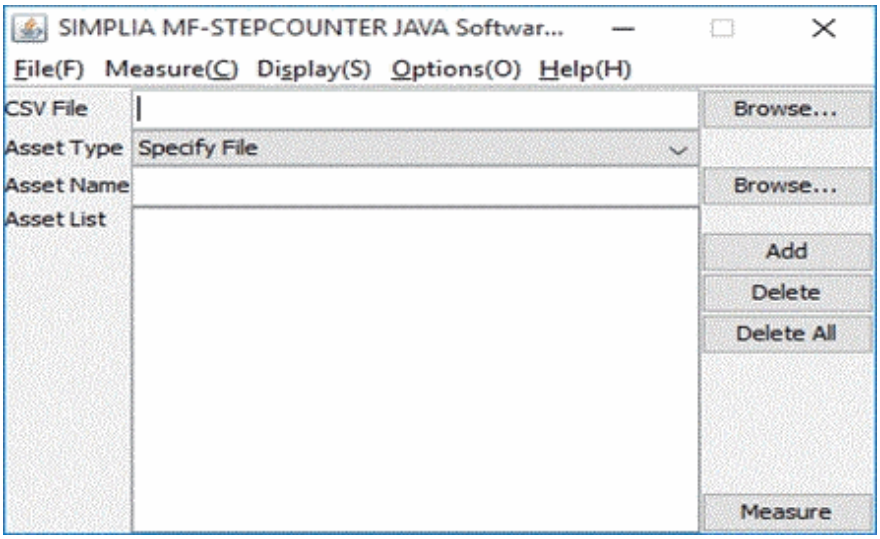
R: Required, P: Possible, D: Disabled, N: Not possible

Chapter 6 User Interface

This chapter explains the dialogs of SIMPLIA MF-STEPCOUNTER Java Software Metrics Measurement.

6.1 Main Dialog

Layout



Description

Specify the following items required for measurement processing.

Required Settings	Description
CSV File	Specify the measurement results output file name.
Asset List	Specify the file name of the measurement target asset (folder/Java file) name.

Description of Menu Commands

Table 6.1 (1) File Menu

Menu Item	Description
New(N)	Performs measurement and creates a new asset file list.
Open Asset File List...(O)	Reads an existing asset file list.
Save(S)	Updates and saves the contents of the asset file list.
Save As...(A)	Saves the asset file list as another file with the specified name.
Exit SIMPLIA MF-STEPCOUNTER(X)	Exits this tool.

Table 6.2 (2) Measure Menu

Menu Item	Description
Measure(K)	Executes measurement processing and displays the measurement results.

Table 6.3 (3) Display Menu

Menu Item	Description
Display Results(D)	Displays the measurement results on screen.

Table 6.4 (4) Options Menu

Menu Item	Description
Specify Class Path...(P)	Specify the search path for class files.
Specify Measurement Options...(I)	Specifies measurement options.
Specify Alarm...(L)	Specifies the display format of the alarm given for the measurement results.

Table 6.5 (5) Help Menu

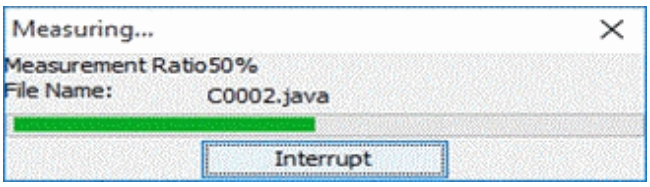
Menu Item	Description
MF-STEPCOUNTER Help(T)	Displays the Help dialog.
Version Information...(A)	Displays the version and level of this tool.

Description of Dialog Controls

Dialog Controls	Description
[CSV File] Text Field	Inputs the CSV file name for the output destination of measurement results. The [Browse] button for [CSV File] is available for file selection.
[Browse] button for [CSV File]	Enables selection of the CSV file name for the output destination of measurement results.
[Asset Type] Combo Box	Enables selection of the type of the target asset.
[Asset Name] Text Field	Inputs the measurement target asset name. Depending on the asset type, folder selection is also supported.
[Browse] button for [Asset Name]	Enables selection of the measurement target asset name. The selection varies depending on the selection in the [Asset Type] combo box. For details, refer to "5.2.2 Target Asset Specification Method."
[Asset List]	Displays the names of assets already selected.
[Add] Button	Appends the string input in the [Asset Name] text field to the [Asset List].
[Delete] Button	Deletes the asset selected in the [Asset List] from this list.
[Delete All] Button	Deletes all assets displayed in [Asset List] from this list.
[Measure] Button	Executes measurement processing and displays the measurement results.

6.2 Interrupt Measurement Dialog Box

Layout



Description

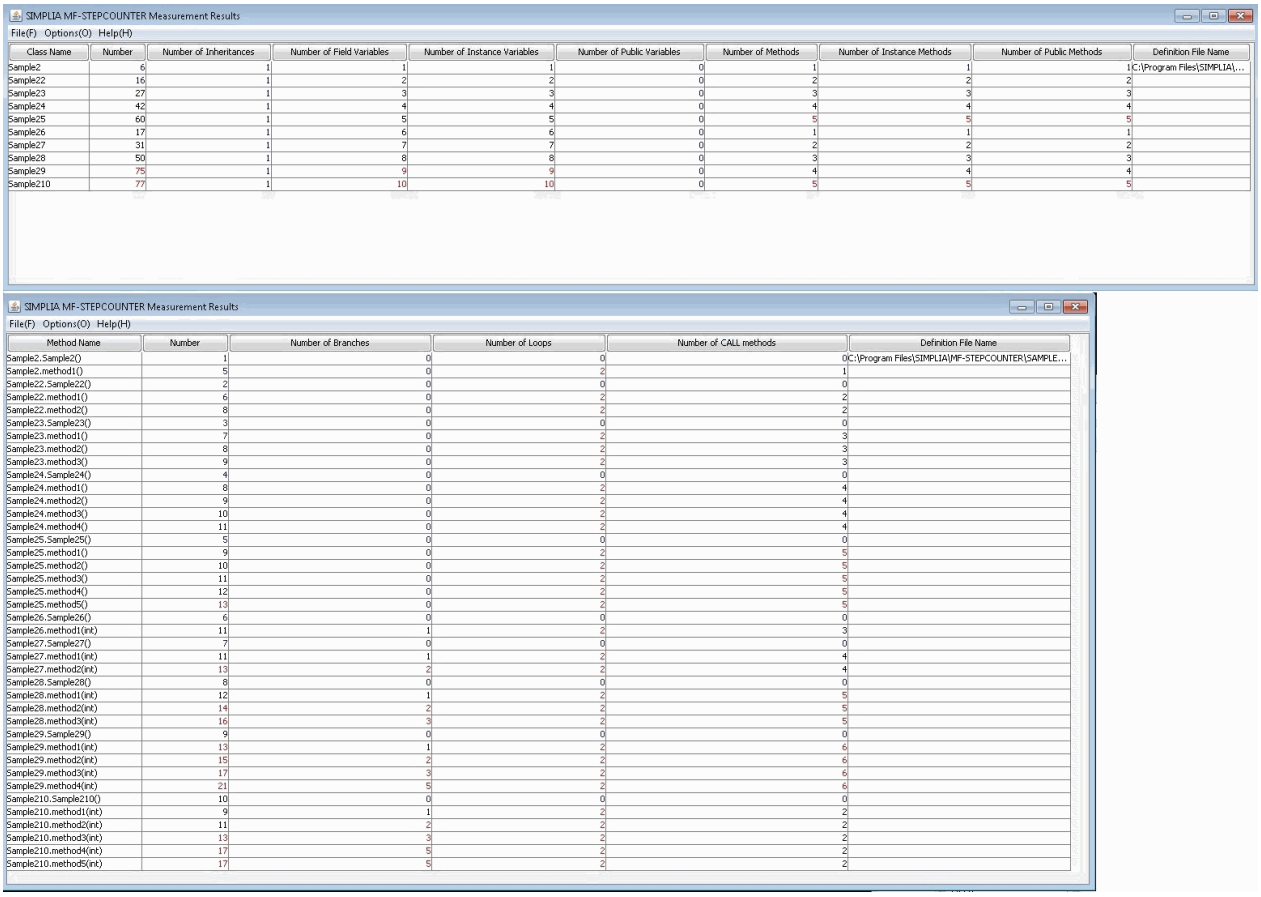
This dialog box displays the measurement status.

Description of Dialog Controls

Dialog Controls	Description
[Measurement Ratio] Label	Displays the measurement ratio of the assets specified in the Main dialog.
[File Name] Label	Displays the file name of the Java file currently being measured.
Measurement Status Canvas	The measurement progress is indicated.
[Interrupt] Button	Interrupts the process after the measurement of the Java file currently being measured is completed.

6.3 Measurement Results Dialog

Layout



Description

This dialog displays the measurement results in table format. The displayed data contents vary depending on the specified measurement unit.

Table 6.6 (1) For Measurement by Class Unit

Displayed Data Item	Description
Class Name	The class name that is the measurement unit is displayed.
Number	The number of statements within the class is displayed.
Number of Inheritances	The number of inheritance classes is displayed.
Number of Field Variables	The number of field variables defined within the class is displayed.

Displayed Data Item	Description
Number of Instance Variables	The number of instance variables defined within the class is displayed.
Number of Public Variables	The number of public variables defined within the class is displayed.
Number of Methods	The number of methods defined within the class is displayed.
Number of Instance Methods	The number of instance methods defined within the class is displayed.
Number of Public Methods	The number of public methods defined within the class is displayed.
Definition File Name	The name of the Java file defining the class is displayed. This is output only for the class name defined first in the Java file.

Table 6.7 (2) For Measurement by Method Unit

Displayed Data Item	Description
Method Name	The method name that is the measurement unit is displayed.
Number	The number of statements within the method is displayed.
Number of Branches	The number of conditional branches defined within the method is displayed.
Number of Loops	The number of loops defined within the method is displayed.
Number of CALL Methods	The number of CALL methods defined within the method is displayed.
Definition File Name	The name of the Java file defining the method is displayed. This is output only for the method name defined first in the Java file.

Description of Menu Commands

Table 6.8 (1) File Menu

Menu Item	Description
Exit(X)	Exits the Measurement Results dialog.

Table 6.9 (2) Options Menu

Menu Item	Description
Specify Alarm...(L)	Specifies the display format of the alarm given for the measurement results.

Table 6.10 (3) Help Menu

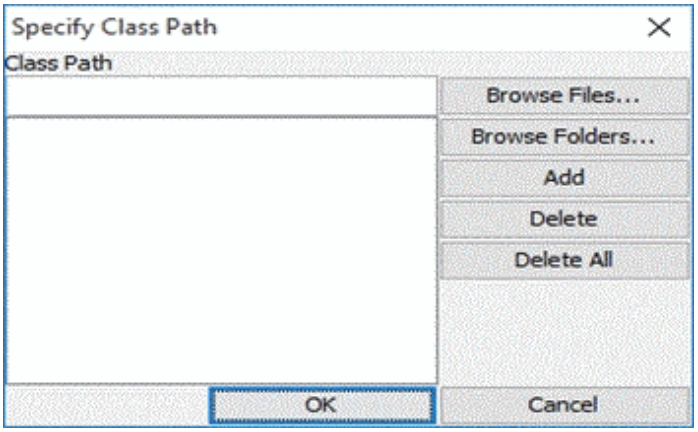
Menu Item	Description
MF-STEPCOUNTER Help(T)	Displays the Help dialog.
Version Information...(A)	Displays the version and level of this tool.

Description of Dialog Controls

Dialog Controls	Description
Measurement Results Table	Displays the measurement results. The results are displayed in the following font colors according to the display format of the alarm identification specified in the Specify Alarm dialog box. - Measurements equal to or above the upper limit: Red - Measurements equal to or below the lower limit: Blue - Values other than those above: Black

6.4 Specify Class Path Dialog Box

Layout



Description

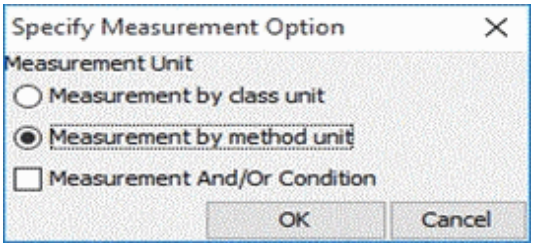
Specify the class path (folder name or JAR file name) to be the search path for class file measurement.

Description of Dialog Controls

Dialog Controls	Description
[Class Path] Text Field	Inputs the folder name or JAR file name to be set as the class path. The [Browse Files] button and the [Browse Folders] button are available to select a JAR file or a folder.
[Browse Files] Button	Select the Jar file name to be set as the class path.
[Browse Folders] Button	Select the folder name to be set as the class path.
Class Path List	Displays the class path names already selected.
[Add] Button	Appends the string input in the [Class Path] text field to the [Class Path List].
[Delete] Button	Deletes the asset selected in the [Class Path List] from this list.
[Delete All] Button	Deletes all paths displayed in the [Class Path List] from the list.
[OK] Button	Saves the settings and closes this dialog box.
[Cancel] Button	Discards the settings and closes this dialog box.

6.5 Specify Options Dialog Box

Layout



Description

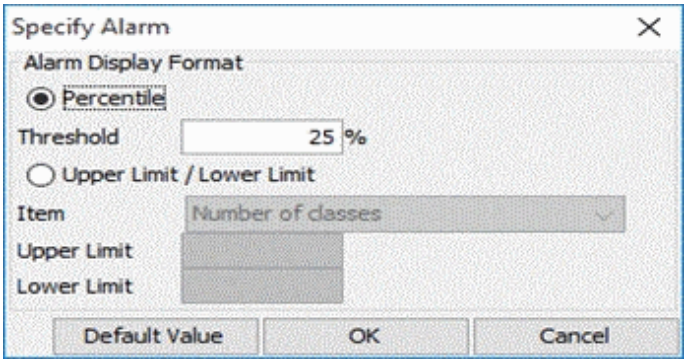
Specify the measurement unit (measurement by class unit or measurement by method unit).

Description of Dialog Controls

Dialog Controls	Description
[Measurement by class unit] Radio Button	Executes the measurement processing in units of the classes defined in the Java file set as the asset.
[Measurement by method unit] Radio Button	Executes the measurement processing in units of the methods defined in the Java file set as the asset.
[Measurement And/Or Condition] Check Box	Specifies to measure the number of branches at each And/Or condition in a statement such as an if statement. This is only enabled for measurement by method unit.
[OK] Button	Saves the settings and closes this dialog box.
[Cancel] Button	Discards the settings and closes this dialog box.

6.6 Specify Alarm Dialog Box

Layout



Description

Specify the range of alarm indication to be given in the measurement results dialog. The measurements above the upper limit and below the lower limit specified in this dialog box are the targets for alarm indication.

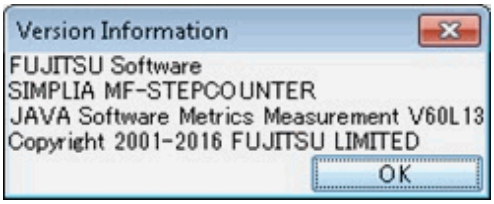
Description of Dialog Controls

Dialog Controls	Description
[Percentile] Radio Button	Gives percentile alarm indication. For details, refer to "4.2.3 Percentile Alarm Indication".
[Threshold] Text Field	Sets the lower threshold based on which alarm indication is given. The lower threshold should be set to a value in the range of (0% to 50%). If this field is left blank, or if 0% is specified, no alarm indication is given. This is enabled only if the [Percentile] radio button is selected.
[Upper Limit / Lower Limit] Radio Button	Gives alarm indication in the upper limit/lower limit format.
[Item] Combo Box	Allows selection of measurement items to be the targets for alarm specification. This is only enabled if the [Upper Limit / Lower Limit] radio button is clicked.
[Upper Limit] Text Field	Sets the upper limit value based on which alarm indication is given. This value must be larger than the lower limit value. If this field is left blank, no alarm indication is given for measurements based on the upper limit. This is only enabled if the [Upper Limit / Lower Limit] radio button is clicked.
[Lower Limit] Text Field	Sets the lower limit value based on which alarm indication is given. This value must be smaller than the upper limit value. If this field is left blank, no alarm indication is given

Dialog Controls	Description
	for measurements based on the lower limit. This is only enabled if the [Upper Limit / Lower Limit] radio button is clicked.
[Default Value] Button	By default, Percentile is selected and Threshold is set to 25%.
[OK] Button	Saves the settings and closes this dialog box.
[Cancel] Button	Discards the settings and closes this dialog box.

6.7 Version Information Dialog Box

Layout



Description

This dialog box displays the tool name, function name, version and level, and copyright.

Description of Dialog Controls

Dialog Controls	Description
[OK] Button	Closes this dialog box.

Chapter 7 Supplementary Notes

This chapter provides supplementary information regarding Java Software Metrics Measurement.

7.1 Advisory Notes

Precautions to be observed when using Java Software Metrics Measurement are as follow.

1. About measurement target Java files/class files

   The target Java file must be a file that is properly compilable and described in Shift_JIS. In addition, the target Java file must be a file created with JDK8.0.

2. About interruption of measurement

   If measurement is interrupted during screen processing, measurements to the point of the interruption are output as the measurement results. If the application is interrupted from the command prompt during batch processing or at the startup of screen processing, no measurement results are output.

3. About multiple starts

   This tool does not support multiple starts. Therefore, this tool is not guaranteed to work in cases where multiple instances of this tool are started.

4. About the active folder on startup

   At the startup of this tool, the current folder must be the installation folder of this tool.

5. About tool configurations

   The installation folder where this tool is installed contains the temp and help subfolders. If these subfolders in the installation folder are changed or deleted, operation of this tool is not guaranteed.

6. About SQLJ-based sources

   This tool does not support the measurement of Java sources based on SQLJ. If a Java source based on SQLJ is targeted for measurement, the source will not be measured.

7. About the path length

   This tool allows the specification of up to 260 bytes in the path to a file to use. Depending on the file system, the longest specifiable path may be less than 260 bytes. In this case, only the allowed number of bytes are used for the path, and the rest of the path is truncated. Caution is necessary when specifying paths with the maximum length.

8. About the enumeration type and comment measurement

   There are the following precautions for JDK 8.0.

   ・About the enumeration type

   If an enum class is declared, two public static methods, valueOf() and values(), are added, and two is added to the number of methods described.

   If an enum class and a method are declared, another one is added the number of methods because an internal method is maintained.

   Even when an enum field is declared in a normal class, the field is separately measured as a class.

   ・About comments

   Step Measurement measures comments from the viewpoint of measuring the quantity of description, but it measures comments as valid lines not comment lines (comments). In contrast, this Metrics Measurement does not include comments in number measurement from the viewpoint of perceiving source complexity/characteristics.

   If the source defining the comment type is measured by method, no measurement results are output because no process exists.

   Sources defining the comment type within a class/interface declaration cannot be measured.

9. About the measurement of Java files defining the same class name

   When multiple Java files defining the same class name (including package names) are specified as measurement targets, classes defined in the measured Java files cannot be measured properly. As class files are automatically searched according to the order of paths specified in the CLASSPATH environment variable/paths specified as the class path in this tool, the same class file is always detected.

   When measuring multiple Java files defining the same class name (including package names), it is necessary to perform the measurement in several batches.

10. About the measurement of class files using a user-defined package

    When measuring class files using classes within a user-defined package, it is necessary to specify the path (folder/JAR file) where the package used is located in the CLASSPATH environment variable in advance. If the path is unspecified, the class files using classes included in a user-defined package cannot be measured properly.

11. About class file and Java file synchronization

    For measurement by class unit, class files must be in synchronization with the relevant Java file and be kept up-to-date. If they are not up-to-date, files cannot be measured properly. Before measurement, confirm that the Java file has already been compiled.

12. About CSV file output

    This tool overwrites the contents of the specified CSV file with the measurement results when it performs measurement. To retain existing information in the specified CSV file, either change the destination CSV file name, or create a backup of the CSV file before performing measurement.

    If an asset file list is specified as a command line argument and measurement is performed in batch mode, the measurement results are output to the CSV file specified in the asset file list. Any existing information in the specified CSV file is also overwritten with the measurement results in this case, so caution is necessary.

13. About Help display frame behavior

    In the Help display frame, if a link destination is selected, any selection is prohibited temporarily while loading the HTML page. If the loading of the HTML page continues even after the selection prohibited status is released, frequent link occurrence may cause the display of Javax.swing exception messages in the command prompt, but there is no problem in the processing.

14. About the operational environment of the JDK

    For this tool to operate, the PATH environment variable must be configured in advance to include the bin folder of JDK.

15. About network drives during file or folder selection

    In the Select File or Select Folder dialog box invoked for CSV file selection, asset file/asset folder selection, or class path file/class path folder selection, direct access to network computers is not possible. To access a network computer, it is necessary to map a network drive using Explorer in advance.

16. About dialog operations

    A continuous operation (continuously depressing the same button) during dialog operations may lead to unexpected results.

17. About the use character

    Please do not use the environment dependent character of UNICODE for the storage path of the JAVA File and the CLASS File.

Chapter 8 Messages

This chapter explains the details of errors output in Java Software Metrics Measurement.

8.1 Message List

STJ-001 JDK7.0 or later is necessary if you want to execute this tool.

[Description]

JDK 8.0 is necessary to execute this tool.

[Corrective Action]

Install JDK 8.0 and then perform the operation again.

STJ-002 The threshold of the specified alarm is invalid.

[Description]

There is an error in the alarm range (threshold) specified in the Specify Alarm dialog.

[Corrective Action]

Modify the alarm range settings.

STJ-003 The upper limit or lower limit of the specified alarm is invalid.

[Description]

There is an error in the alarm range (upper limit/lower limit) specified in the Specify Alarm dialog.

[Corrective Action]

Modify the alarm range settings.

STJ-004 The CSV file name has not been specified.

[Description]

The CSV file name has not been specified in the Main dialog.

[Corrective Action]

Specify the CSV file name.

STJ-005 The class or inheritance class of the target cannot be found.
JAVA=[java file name]
CLASS=[class name]

[Description]

During class file measurement, the class or inheritance class of the target cannot be detected.

[Corrective Action]

Set the path for the class or inheritance class of the target using Specify Class Path of this tool.

Example 1: When the class name is aaa.class and the package is com.xxx.yyy, and the folder structure is as follows

```
c:\classes\com\xxx\yyy\aaa.class -> Specify c:\classes in folder selection
```

Example 2: When aaa.class is compressed in a JAR file and located as follows

```
c:\classes\jar\abc.jar -> Specify c:\classes\jar\abc.jar in file selection
```

Example 3: When aaa.class inherits the com.xxx.bbb class

```
c:\classlib\com\xxx\bbb.class -> Also specify c:\classlib in folder selection
```

Note: The folder structure in a situation where files are expanded to folders and the folder structure within a JAR file must be consistent with that of the package as shown in example 1 above, as with classpath specification in the normal Java Runtime Environment.

In addition, if the length of the full path to a class file or JAR file exceeds the upper limit on the OS, this error may occur due to the inability to access the file.

STJ-006 The specification of the class path is invalid.

[Description]

There is an error in the class path specified in the Specify Class Path dialog.

[Corrective Action]

Modify the content of the set class path.

STJ-007 The same class path is already specified.

[Description]

The same class path has already been specified in the Specify Class Path dialog.

[Corrective Action]

The class path is already included in the class path list, so no specification is necessary.

STJ-008 An error occurred during measurement.

[Description]

An error occurred during measurement.

[Corrective Action]

Check the error messages sent to the error file "mfstpjm.err", which is output to the temp folder in the installation folder of this tool, and take corrective action for each message.

STJ-009 Interrupt processing.
Are you sure?

[Description]

During measurement, the [Interrupt] button was clicked in the Interrupt dialog box.

[Corrective Action]

Click the [Yes] button to interrupt the measurement, or click the [No] button not to interrupt.

STJ-010 Measurement complete.

[Description]

Measurement was completed successfully.

[Corrective Action]

None.

STJ-011 Failed to read the specified file.
TYPE=[file type]
FILE=[file name]

[Description]

Reading of the specified file failed.

[Corrective Action]

Confirm that the specified file is a readable file of the specified file type. If necessary, modify the specified file name and try again.

STJ-012 Failed to write to the specified file.
TYPE=[file type]
FILE=[file name]

[Description]

Writing to the specified file failed.

[Corrective Action]

Confirm that the specified file is a writable file of the specified file type. If necessary, modify the specified file name and try again.

STJ-013 Displaying the measurement results.
Please close the measurement result frame.

[Description]

While the Measurement Results dialog is displayed, the selected process cannot be executed.

[Corrective Action]

Close the Measurement Results dialog and perform the operation again.

STJ-014 Failed to display error information.
Please refer to the standard output or the error file (mfstpjm.err).

[Description]

Display of the Error Information display dialog failed.

[Corrective Action]

Set the error output destination to standard output, and perform measurement again, or check the contents of the error file "mfstpjm.err", which is output to the temp folder in the installation folder of this tool.

STJ-015 An error occurred during measurement.
Display the error file?

[Description]

An error occurred during measurement.

[Corrective Action]

Click the [Yes] button to view the contents of the error file in the Error Information display dialog, or click the [No] button not to.

STJ-016 Java file format is incorrect.
FILE=[Java file name]
ERR=[location of the error]

[Description]

The format of the specified Java file is incorrect.

[Corrective Action]

Check the location of the error in the Java file and correct the error. The specified Java file must be a file that can be compiled properly.

STJ-017 There are no Java files in the measurement target.
PATH=[asset Name]

[Description]

There are no Java files to measure in the specified measurement target.

[Corrective Action]

Specify the name of an asset in which measurement target Java files exist.

STJ-018 Failed to read the file.
TYPE=[file type]
FILE=[file name]

[Description]

Reading of the specified file failed.

[Corrective Action]

Confirm that the specified file name points to a writable file of the correct file type.

STJ-019 Failed to write to the file.
TYPE=[file type]
FILE=[file name]

[Description]

Writing to the specified file failed.

[Corrective Action]

Confirm that the specified file name points to a readable file of the correct file type.

STJ-020 The file format is incorrect.
TYPE=[file type]
FILE=[file name]

[Description]

The format of the specified file does not correspond to the specified file type.

[Corrective Action]

Confirm that the specified file name points to a file of the correct file type.

STJ-021 The file does not exist.
TYPE=[file type]
FILE=[file name]

[Description]

The specified file does not exist.

[Corrective Action]

Specify the name of an existing file of the correct file type.

STJ-022 Field information cannot be found.
JAVA=[Java file name]
CLASS=[class name]

[Description]

The tool failed to detect field information during class file measurement.

[Corrective Action]

Check whether a user-defined package has been used within the target class. If a user-defined package has been used, specify the path (folder/JAR file) where that package is located through class path specification of the tool. If not specifying, confirm that the target class file is a normal file.

STJ-023 Method information cannot be found.
JAVA=[Java file name]
CLASS=[class name]

[Description]

The tool failed to detect method information during class file measurement.

[Corrective Action]

Check whether a user-defined package has been used within the target class. If a user-defined package has been used, specify the path (folder/JAR file) where that package is located through class path specification of the tool. If not specifying, confirm that the target class file is a normal file.

STJ-024 The same asset name is already specified.

[Description]

The same asset name is already specified in the Main dialog.

[Corrective Action]

The asset name specified in the Main dialog is already included in the Asset List, so no specification is necessary. If the asset type is a folder containing subfolders, modify the specification to designate the higher level folder.

STJ-025 The asset does not exist.

[Description]

The asset specified in the Main dialog does not exist.

[Corrective Action]

Specify an existing asset.

STJ-026 The asset does not exist.
OBJECT=[name of an asset]

[Description]

The asset specified in the Main dialog does not exist.

[Corrective Action]

Specify an existing asset.

STJ-027 The asset name is unspecified.

[Description]

The name of the asset to be added to the Asset List has not been specified in the Main dialog.

[Corrective Action]

Specify the name of an existing asset, and add it to the list.

STJ-028 The format for the asset has been changed.
Selection of the asset list will be deleted.
continue?

[Description]

The asset type has been changed in the Main dialog. The asset name specified in the Asset List will deleted from the list due to an inconsistency in the format.

[Corrective Action]

Click the [Yes] button to delete the asset name from the Asset List and change the asset type. Click the [No] button to retain the asset name in the Asset List and not to change the asset type.

STJ-029 The format of the asset is invalid.

[Description]

The asset type specified for the asset name in the Main dialog is inconsistent with the type selected for the asset type.

[Corrective Action]

Specify the name of an asset whose type is consistent with the specified asset type.

STJ-030 The format of the asset is invalid. OBJECT=[name of an asset]

[Description]

The asset type specified for the asset name in the Main dialog is inconsistent with the type selected for the asset type.

[Corrective Action]

Specify the name of an asset whose type is consistent with the specified asset type.

STJ-031 The command line parameter is invalid.

[Description]

The command line parameter specified at startup of this tool is invalid.

[Corrective Action]

Modify the command line parameter and perform the operation again.

STJ-032 The files of the asset list have not been saved. Do you want to save them?

[Description]

The contents specified in the Main dialog/Specify Class Path dialog box/Specify Measurement Options dialog box/Specify Alarm dialog box have not been saved.

[Corrective Action]

Click the [Yes] button to save the settings, or click the [No] button not to save.

STJ-033 The update date of the class file is older than the Java source. JAVA=[Java file name] CLASS=[class name]

[Description]

The class file was created before the target Java file was updated.

[Corrective Action]

Compile the target Java file and confirm that it is in synchronization with the created class file. In addition, set the class file search path using Specify Class Path of this tool to confirm that it is valid.

STJ-034 Failed to output the error file. Error messages will be output using standard output.

[Description]

Output of the error file (mfstpjm.err) failed. Error messages generated during measurement will be output to standard output.

[Corrective Action]

Confirm that the error file (mfstpjm.err) in the temp folder in the installation folder of this tool is a file to which error messages can be output.

STJ-035 Unable to refer to the bin folder of JDK.

[Description]

Reference to the bin folder in the JDK installation folder failed.

[Corrective Action]

Add the bin folder of the JDK to the PATH environment variable and perform the operation again.

STJ-036 The specified path already exists. Overwrite it?

[Description]

The specified path already exists.

[Corrective Action]

Click the [Yes] button to overwrite it, or click the [No] button not to overwrite.

STJ-037 The specified path does not exist. Please specify an existing path.

[Description]

The specified path does not exist.

[Corrective Action]

Re-specify using a path that exists.

STJ-038 Failed to obtain message.

[Description]

Failed to get the message to be displayed.

[Corrective Action]

This tool may not have been installed successfully. Perform the operation again, after installation.

Chapter 9 Appendix

This appendix explains the output results and samples of Java Software Metrics Measurement.

9.1 Description of Measurement Results (CSV Files)

Overview

Measurements are output to a CSV file. If measurement is interrupted using the Interrupt button during screen processing, measurements to the point of the interruption are output. The contents output to the CSV file vary depending on the measurement unit.

Description of Output Contents

(1) For Measurement by Class Unit

```
"class-name" , 999 , 999 , 999 , 999 , 999 , 999 , 999 , 999 , "file-name"
       a         b     c     d     e     f     g     h     i         j
```

Section	Measurement Item	Description
a	Class Name	The target class name is output in the "Package Name.Class Name" format.
b	Number	The number of statements within the class is output.
c	Number of Inheritances	The number of inheritance classes is output.
d	Number of Field Variables	The number of field variables defined within the class is output.
e	Number of Instance Variables	The number of instance variables defined within the class is output.
f	Number of Public Variables	The number of public variables defined within the class is output.
g	Number of Methods	The number of methods defined within the class is output.
h	Number of Instance Methods	The number of instance methods defined within the class is output.
i	Number of Public Methods	The number of public methods defined within the class is output.
j	Definition File Name	The full path to the Java file defining the class is output. This is output only for the class defined first in the Java file.

(2) For Measurement by Method Unit

```
"method-name" , 999 , 999 , 999 , 999 , "file-name"
       a           b     c     d     e        f
```

Section	Measurement Item	Description
a	Method Name	The target method name is output in the "Package Name.Class Name.Method Name (Argument)" format.
b	Number	The number of statements within the method is output.
c	Number of Branches	The number of conditional branches within the method is output.
d	Number of Loops	The number of loops within the method is output.
e	Number of CALL Methods	The number of CALL methods within the method is output.

f	Definition File Name	The full path to the Java file defining the method is output. This is output only for the method defined first in the Java file.

9.2 Sample Usage

In the installation folder of MF-STEPCOUNTER, the folder "SAMPLE\en" is created and sample sources are stored.

File Name	Description
Sample2.java	The sample applet source
SAMPLE2.jar	The JAR file storing the class files generated from Sample2.java

9.2.1 To Measure the Sample Source

1. Start Java Software Metrics Measurement from the Windows Start menu, or select the [New] command on the [File] menu of Java Software Metrics Measurement if it is already running.
2. In the Main dialog, click the [Browse] button next to [CSV File], and specify the CSV file name for measurement results output from the [Select File] dialog box.
3. In the Main dialog, select [Specify File] from the [Asset Type] list.
4. In the Main dialog, click the [Browse] button next to [Asset Name], and specify the "Sample2.java" file located in the "SAMPLE\en" folder from the [Select File] dialog box.
5. In the Main dialog, click the [Add] button to add the file to [Asset List].
6. Register the location of class files corresponding to the measurement target "Sample2.java" as the class path using either of the following methods.

   a. Addition to the CLASSPATH environment variable

   b. Addition from the [Specify Class Path] dialog box accessed from the [Specify Class Path] command on the [Options] menu

   When using method b, click the [Browse Files] button in the [Specify Class Path] dialog box, specify the "SAMPLE2.jar" file located in the "SAMPLE\en" folder from the [Select File] dialog box, and add the file location as the class path with the [Add] button.

   The class paths of classes (such as JDK) referred to from the asset must be added to the CLASSPATH environment variable in advance as well.
7. In the Main dialog, select the [Measure] command from the [Measure] menu.
8. The measurement results are saved in the specified CSV file and displayed on screen.

For details, refer to the following sections.

- 5.2.1 New Asset File List Specification Method
- 5.2.2 Target Asset Specification Method
- 5.2.4 Class Path Specification Method
- 9.1 Description of Measurement Results (CSV Files)

9.2.2 To Change the Measurement Method

[Measurement by class unit] is the default in the initial settings, but changing the setting to [Measurement by method unit] measures classes in more detail.

1. In the Main dialog, select the [Specify Measurement Options] command from the [Options] menu.
2. From the [Specify Measurement Options] dialog box, select [Measurement by method unit].
3. In the Main dialog, select the [Measure] command from the [Measure] menu.
4. The measurement results are saved in the specified CSV file and displayed on screen.

For details, refer to the following sections.

- 4.1.1 Measurement by Class Unit
- 4.1.2 Measurement by Method Unit

9.2.3 To Use Alarm Indication

Specify a percentile threshold or specify the upper limit/lower limit of each measurement item to enable alarm indication in the Measurement Results dialog.

- Alarm indication is given in **red** if the value is equal to or above the upper limit
- Alarm indication is given in **blue** if the value is equal to or below the lower limit

[Percentile] Specification

1. In the Main dialog, select the [Specify Alarm] command on the [Options] menu.
2. Click the [Percentile] radio button in the [Specify Alarm] dialog box.
3. Set the threshold.
4. In the Main dialog, select the [Measure] command on the [Measure] menu to perform measurement again, or select the [Display Results] command on the [Display] menu to redisplay the measurement results.
5. The [Specify Alarm] command on the [Options] menu in the Measurement Results dialog can also be used as an alternative to this specification method.

[Upper Limit / Lower Limit] Specification

1. In the Main dialog, select the [Specify Alarm] command on the [Options] menu.
2. Click the [Upper Limit / Lower Limit] radio button in the [Specify Alarm] dialog box.
3. Select an item from the [Item] list.
4. Set the upper limit and the lower limit.
5. In the Main dialog, select the [Measure] command on the [Measure] menu to perform measurement again, or select the [Display Results] command on the [Display] menu to redisplay the measurement results.
6. The [Specify Alarm] command on the [Options] menu in the Measurement Results dialog can also be used as an alternative to this specification method.

For new measurements, [Percentile] and [Threshold = 25%] are the default values.

When not using alarm indication, leave Threshold and Upper Limit / Lower Limit blank (with no data input).

9.2.4 To Save the Target

The specified measurement target and output destination CSV file, including various option settings, can be saved as an [Asset File List].

1. In the Main dialog, select the [Save As] command from the [File] menu.
2. In the [Save as] dialog box, specify the destination to save.
3. In the [Save as] dialog box, click the [Save] button to save the measurement target.

9.2.5 To Perform Remeasurement

Read a previously saved [Asset File List] to perform remeasurement simply.

1. In the Main dialog, select the [Open Asset File List] command on the [File] menu.
2. Specify the file to be read in the [Open File] dialog box.
3. In the [Open File] dialog box, click the [Open] button to read the file.
4. In the Main dialog, select the [Measure] command from the [Measure] menu.

9.2.6 About Judgment of Measurement Results

For details on how to judge the measurement results, refer to "4.2.1 Judgment on Measurements."

For details on how to configure alarm criteria settings, refer to "4.2.2 Alarm Criteria Configuration Examples."