cc gmbh

CC ANALYZER

CC Reengineering
CC Open Source
CC Outsourcing
CC Consulting
CC Training
CC ANALYZER
CC AUDITOR
CC ARTISAN
CC ASSESS
CC OPAIRA

Test Coverage Monitor
für C, C++, COBOL, Java, JSP und PL/I

cc gmbh

CC ANALYZER
Test Coverage Monitor
für C, C++, COBOL, Java, JSP und PL/I

Kurzbeschreibung:

CC ANALYZER misst die Testabdeckung von C/C++-, COBOL-, Java-, JSP- und PL/I-Programmen und dient der dynamischen Qualitätssicherung in der Softwareentwicklung und Programmwartung.
Dynamisches und systematisches Testen steigert die Qualität der Anwendungen. Somit vermeidet der Einsatz von **CC ANALYZER** für C/C++, COBOL, Java, JSP und PL/I weitgehend viele der üblichen Probleme bei der Produktionseinführung neuer oder geänderter Anwendungen.

Dialogsprache	Deutsch, Englisch
Anwendungsgebiete	Softwareentwicklung Softwarewartung Qualitätssicherung Softwaresicherheit e-Commerce Tuning Testumgebung
Sprachen	C/C++ COBOL Java JSP PL/I
Systemumgebung	Windows IBM z/OS UNIX AIX

Sie sind der Meinung, dass es keine 100%ige Garantie gibt? Überlegen Sie noch einmal. Mit **CC ANALYZER** steht Ihnen ein Tool für einen Anwendungstest zur Verfügung, bevor Ihre Anwendung in Produktion geht. Ist **CC ANALYZER** ein Teil des Testprozesses, heißt es nicht "probieren wir es aus". Der Übergang erfolgt stattdessen gesichert und mit minimalen manuellen Input.

CC ANALYZER ist ein Testabdeckungswerkzeug – sicher, effizient, exakt und umfassend. **CC ANALYZER** arbeitet automatisch und eignet sich ausgezeichnet als Qualitätsanalyse-Werkzeug.

CC ANALYZER für C/C++, COBOL, Java, JSP und PL/I

- liefert exakte und logisch strukturierte Methoden für den Test der Anwendungen
- organisiert und optimiert alle Informationen über die Testzyklen
- eignet sich zur dynamischen Qualitätssicherung
- in der Softwareentwicklung und Programmwartung

Das systematische Testen definiert und setzt automatisch hohe Qualitätsstandards und führt direkt zur drastischen Reduzierung der Probleme, welche oft bei einer Produktionseinführung von neuen oder überholten Anwendungen auftreten.

Der Bedarf an Sicherheit und Leistungsfähigkeit beim Überwachen der Tests nimmt ständig zu. Neue Märkte, e-Business, e-Commerce und damit mehr Kundennähe verursachen eine dynamische Veränderung der Entwicklungs- und Produktionsumgebungen. Die Kosten, das Zeitmanagement und die Ressourcen sind die Hauptfaktoren für den wirtschaftlichen Erfolg.

Und das bestimmt letztlich, worauf sich Unternehmen konzentrieren sollten:

- bessere Qualität und kürzere Testzyklen
- objektive Festlegung der Testprozesse
- übersichtliche Darstellung der Testergebnisse
- zuverlässige Dokumentation über die Testläufe

cc gmbh

CC ANALYZER
Test Coverage Monitor
für C, C++, COBOL, Java, JSP und PL/I

Funktionsmerkmale

CC ANALYZER ist ein Qualitätssicherungs-Werkzeug zur Messung der Testabdeckung von Anwendungen im Rahmen der Software-entwicklung und Programmwartung.

Ausgewählte Funktionsmerkmale sind:

Identifizierung nicht getesteter Programme

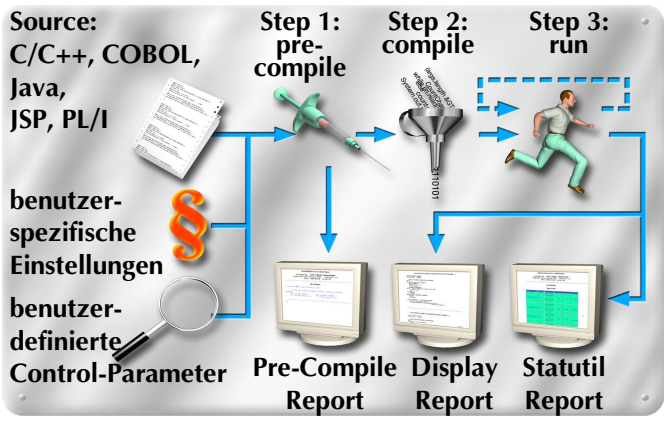
- Umfassende Messung der ausgeführten Programme in Entwicklungs-, Tests- und Produktionsumgebungen
- Bewertung der Vollständigkeit der Testläufe durch Anzeige der Anzahl der durchlaufenen Intervalle eines Programms

Identifizierung nicht getesteter Codes

- Identifizierung nicht durchlaufener Programmteile in einem Testlauf
- Bestimmt die Häufigkeit der ausgeführten Kommandos

Steigerung der Qualität und Optimierung der Testdaten

- Zusammenstellung der benötigten Informationen für qualitative Testdaten und Optimierung der bestehenden Daten
- Wesentliche Steigerung der Qualität von Testdaten während des Entwicklungs- und Modifikations-/Verbesserungsprozesses



Testreihen spezifizieren

- Analysiert vollständige Systeme oder individuell ausgewählte Programme und Intervalle
- gezieltes Testen der Testabdeckung während Wartungsarbeiten

Benutzerspezifische Automation

- Ausführen spezifischer automatischer Modifikationen, z. B. zusätzlicher Debugging- oder Trace-Funktionen
- Automatisches Einfügen von Kommentaren

Übersichtliche Ergebnisdarstellung im HTML- und Textformat

- Der Report zeigt die absolute sowie die prozentuale Anzahl der durchlaufenen Testintervalle

Was kann **CC ANALYZER** noch?

- **CC ANALYZER** für Java analysiert alle Java-Klassen (Applets, Servlets, Beans, etc.)
- Liefert verläßliche Aussagen über die Qualität von Testdaten und Testläufen
- Trägt zur Testdatenoptimierung bei
- Besitzt zusätzlich eine Trace-Funktion zur einfachen Lokalisierung von Fehlern durch gezielte Testläufe
- Ermöglicht durch qualifizierte Testlaufinfor-mationen mit der Trace-Funktionalität die schnelle Einarbeitung in Fremdprogramme
- Deckt Überschneidungen von Testläufen auf
- Zeigt die Testabdeckung durch einen oder mehrere Testläufe auf
- Erleichtert das Erkennen von Programmab-läufen durch "maßgeschneiderte" Testdaten
- Hilft bei der Verringerung unproduktiver Testläufe durch qualifizierte Analyse-ergebnisse
- Bietet eine zuverlässige Testdokumentation mit der Möglichkeit der Synchronisation der vordefinierten Test-Ende-Kriterien
- Vollständige Integration in Eclipse

CC ANALYZER für C – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-06-30 16:16:15

Display Summary

Source Input File: count.c
Statistics File: statfile_c.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-06-30-16.07.47.00 / 2011-06-30-16.07.47.00	7	2	28.57 %
Test 2	2011-06-30-16.10.11.00 / 2011-06-30-16.10.11.00	7	6	85.71 %
TOTAL	2011-06-30-16.07.47.00 / 2011-06-30-16.10.11.00	7	7	100.00 %

CC ANALYZER für C - Report for Test Number: 1

Source Input File: count.c
Statistics File: statfile_c.dat

```
        void CountChars (FILE *ptr) {
********    char ch;
*       *    int count = 0;
               while (!feof(ptr)) {
********       ch = fgetc(ptr);
               if (!feof(ptr))
********          count++;
            }
********    printf("No. of chars counted = %d\n", count);
         }
         void main(int argc, char *argv[]) {
(000001)    FILE *fptr;
            char filename[100];
            if (argc > 1) {
********      strcpy(filename, argv[1]);
*       *      fptr = fopen(filename, "r");
*       *      CountChars(fptr);      fclose(fptr);
            }
            else
(000001)       printf("Enter the file name as arg to count chars\n");
         }
```

```
Out of    7 Intervals,     2 ( 28.57%) were executed
     1 Test Run(s) represented from 2011-06-30 16.07.47.00
                                 to 2011-06-30 16.07.47.00
```

CC ANALYZER für C - Report for Test Number: 2

Source Input File: count.c
Statistics File: statfile_c.dat

```
         void CountChars (FILE *ptr) {
(000001)    char ch;
            int count = 0;
               while (!feof(ptr)) {
(000722)       ch = fgetc(ptr);
               if (!feof(ptr))
(000721)          count++;
            }
(000001)    printf("No. of chars counted = %d\n", count);
         }
         void main(int argc, char *argv[]) {
(000001)    FILE *fptr;
            char filename[100];
            if (argc > 1) {
(000001)      strcpy(filename, argv[1]);
              fptr = fopen(filename, "r");
              CountChars(fptr);      fclose(fptr);
            }
            else
********       printf("Enter the file name as arg to count chars\n");
         }
```

```
Out of    7 Intervals,     6 ( 85.71%) were executed
     1 Test Run(s) represented from 2011-06-30 16.10.11.00
                                 to 2011-06-30 16.10.11.00
```

Accumulated Results of Test Runs

Source Input File: count.c
Statistics File: statfile_c.dat

```
         void CountChars (FILE *ptr) {
(000001)    char ch;
            int count = 0;
              while (!feof(ptr)) {
(000722)       ch = fgetc(ptr);
               if (!feof(ptr))
(000721)          count++;
            }
(000001)    printf("No. of chars counted = %d\n", count);
         }
         void main(int argc, char *argv[]) {
(000002)    FILE *fptr;
            char filename[100];
            if (argc > 1) {
(000001)       strcpy(filename, argv[1]);
               fptr = fopen(filename, "r");
               CountChars(fptr);      fclose(fptr);
            }
            else
(000001)       printf("Enter the file name as arg to count chars\n");
         }
```

```
Out of    7 Intervals,     7 (100.00%) were executed
     2 Test Run(s) represented from 2011-06-30 16.07.47.00
                                 to 2011-06-30 16.10.11.00
```

CC ANALYZER für COBOL - HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-07-22 10:03:33

Display Summary

Source Input File: DEMO1.COB
Statistics File: statfile_cobol.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-07-22 10.02.52.65 / 2011-07-22 10.03.01.18	25	14	56.00 %
Test 2	2011-07-22 10.03.06.06 / 2011-07-22 10.03.28.87	25	22	88.00 %
TOTAL	2011-07-22 10.02.52.65 / 2011-07-22 10.03.28.87	25	25	100.00 %

CC ANALYZER for COBOL - Report for Test Number: 1

Source Input File: DEMO1.COB
Statistics File: statfile_cobol.dat

```
        003050*
        003051 PARA-2.
(000001)003052     DISPLAY 'Enter Text to display in subprogram :'
        003053     ACCEPT PARM
        003054     CALL CALL-DEMO2 USING PARM.
        003055
        003100
        003200 PARA-2-X.
(000001)003300     EXIT.
        003400*
        003500 PARA-3.
********003600     MOVE SPACE TO PARM
*       *003700     CALL CALL-DEMO2 USING PARM.
        003800 PARA-3-X.
********003900     EXIT.
*       *004000*
        004100 PARA-4.
********004200     DISPLAY 'http://www.cc-gmbh.de'.
        004300 PARA-4-X.
```

```
Out of   25 Intervals,     14 ( 56.00% ) were Executed
        329 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.02.52.65
                                 to 2011-07-22 10.03.01.18
```

```
                CC ANALYZER for COBOL - Report for Test Number: 2

Source Input File:   DEMO1.COB
Statistics File:     statfile_cobol.dat

-------------------------------------------------------------------------------

         003050*
         003051 PARA-2.
********003052      DISPLAY 'Enter Text to display in subprogram :'
*       *003053      ACCEPT PARM
*       *003054      CALL CALL-DEMO2 USING PARM.
*       *003055
*       *003100
         003200 PARA-2-X.
********003300      EXIT.
*       *003400*
         003500 PARA-3.
(000001)003600      MOVE SPACE TO PARM
         003700      CALL CALL-DEMO2 USING PARM.
         003800 PARA-3-X.
(000001)003900      EXIT.
         004000*
         004100 PARA-4.
(000001)004200      DISPLAY 'http://www.cc-gmbh.de'.
         004300 PARA-4-X.

-------------------------------------------------------------------------------

Out of   25 Intervals,     22 ( 88.00% ) were Executed
         329 Line(s) Read from Source File
       1 Test Run(s) Represented from 2011-07-22 10.03.06.06
                                   to 2011-07-22 10.03.28.87
```

```
                        Accumulated Results of Test Runs

Source Input File:   DEMO1.COB
Statistics File:     statfile_cobol.dat

-------------------------------------------------------------------------------

         003050*
         003051 PARA-2.
(000001)003052      DISPLAY 'Enter Text to display in subprogram :'
         003053      ACCEPT PARM
         003054      CALL CALL-DEMO2 USING PARM.
         003055
         003100
         003200 PARA-2-X.
(000001)003300      EXIT.
         003400*
         003500 PARA-3.
(000001)003600      MOVE SPACE TO PARM
         003700      CALL CALL-DEMO2 USING PARM.
         003800 PARA-3-X.
(000001)003900      EXIT.
         004000*
         004100 PARA-4.
(000001)004200      DISPLAY 'http://www.cc-gmbh.de'.
         004300 PARA-4-X.

-------------------------------------------------------------------------------

Out of   25 Intervals,     25 ( 100.00% ) were Executed
         329 Line(s) Read from Source File
           4 Record(s) Read from Statistics File
           2 Statistics Record(s) Pertain to This Program
       2 Test Run(s) Represented from 2011-07-22 10.02.52.65
                                   to 2011-07-22 10.03.28.87
```

```
                    CC ANALYZER für COBOL - STATISTICS-Report

Licenced by:  CASE CONSULT DEVELOPMENT

(C) CC GmbH 2011                                Version: 5.4
Date: 2011-06-30  16:16:15


  PROGRAM:  DEMO2

              RUN DATE         INTERVALS EXECUTED PERCENT INSTRUMENTATION DATE
         --------------------- --------- -------- ------- ---------------------
         2011-06-30(13:48:17)         5        4      80%   2011-06-30(11:52:50)
         2011-06-30(13:53:36)         5        2      40%   2011-06-30(11:52:50)


------------------------------- REPORT TOTALS ---------------------------------

  PROGRAM    RUNS        RUN DATE          INTERVALS   EXECUTED   PERCENT
  --------  -----  --------------------  ---------  ---------  ---------
  DEMO2         2        2011-06-30             5          5       100%
  --------  -----                        ---------  ---------  ---------
  TOTAL:        2                               5          5       100%

101 - REPORT COMPLETED
```

```
          CC ANALYZER für COBOL - DISPLAY-REPORT FOR TEST NUMBER:    1

SOURCE INPUT FILE:   DEMO2
STATISTICS FILE:     STATFILE


                PROCEDURE DIVISION USING PARM1, PARM2.
                PARA-1.
(000001)            MOVE 'DEMO2' TO PARM1
                    MOVE 'Y' TO MAIN-ENTRY-SW
                    MOVE +1234 TO PARM2.
                    IF PARM2 = 9999
(000001)                DISPLAY 'PARM2 HAS AN IMPOSSIBLE VALUE'
                        GO TO PARA-EXIT
                    END-IF
********            EXIT PROGRAM.
                PARA-EXIT.
(000001)            DISPLAY 'ABORTING PROGRAM'.
                PARA-EXIT-X.
(000001)            EXIT PROGRAM.

OUT OF    5 INTERVALS,    4 ( 80%) WERE EXECUTED.
         89 RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(13:48:17.35)
                                     TO 2011-06-30(13:48:18.89)
```

```
          CC ANALYZER für COBOL - DISPLAY-REPORT FOR TEST NUMBER:    2

SOURCE INPUT FILE:   DEMO2
STATISTICS FILE:     STATFILE


                 PROCEDURE DIVISION USING PARM1, PARM2.
                 PARA-1.
(000010)             MOVE 'DEMO2' TO PARM1
                     MOVE 'Y' TO MAIN-ENTRY-SW
                     MOVE +1234 TO PARM2.
                     IF PARM2 = 9999
********               DISPLAY 'PARM2 HAS AN IMPOSSIBLE VALUE'
*       *              GO TO PARA-EXIT
                     END-IF
(000010)             EXIT PROGRAM.
                 PARA-EXIT.
********             DISPLAY 'ABORTING PROGRAM'.
*       *        PARA-EXIT-X.
********             EXIT PROGRAM.

OUT OF    5     INTERVALS,    2 ( 40%) WERE EXECUTED.
         89     RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(13:53:36.05)
                                     TO 2011-03-30(13:53:37.49)
```

```
          CC ANALYZER für COBOL - ACCUMULATED RESULTS OF TEST RUNS

 SOURCE INPUT FILE:   DEMO2
 STATISTICS FILE:     STATFILE


                  PROCEDURE DIVISION USING PARM1, PARM2.
                  PARA-1.
 (000011)             MOVE 'DEMO2' TO PARM1
                      MOVE 'Y' TO MAIN-ENTRY-SW
                      MOVE +1234 TO PARM2.
                      IF PARM2 = 9999
 (000001)                 DISPLAY 'PARM2 HAS AN IMPOSSIBLE VALUE'
                          GO TO PARA-EXIT
                      END-IF
 (000010)             EXIT PROGRAM.
                  PARA-EXIT.
 (000001)             DISPLAY 'ABORTING PROGRAM'.
                  PARA-EXIT-X.
 (000001)             EXIT PROGRAM.

 OUT OF    5    INTERVALS,    5 (100%) WERE EXECUTED.
          89    RECORDS READ FROM SOURCE FILE
           2 TEST RUN(S) REPRESENTED FROM 2011-06-30(13:48:17.35)
                                      TO 2011-06-30(13:53:37.49)
```

CC ANALYZER für C++ – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-07-22 10:06:58

Display Summary

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-07-22 10.06.23.00 / 2011-07-22 10.06.38.00	16	12	75.00 %
Test 2	2011-07-22 10.06.42.00 / 2011-07-22 10.06.55.00	16	12	75.00 %
TOTAL	2011-07-22 10.06.23.00 / 2011-07-22 10.06.55.00	16	14	87.55 %

CC ANALYZER für C++ - Report for Test Number: 1

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

```
          if (a <= 10)                                    // IF without brace
********     cout << "Number is less than 10";
*      *                                                  // End of IF without brace
          else if ((a >= 10)
                   && (a <= 100))                         // ELSE IF without brace
********     cout << "Number is between 10 and 100";
*      *                                                  // End of ELSE IF without brace
          else                                            // ELSE without brace
(000001)     cout << "Number is greater than 100";
                                                          // ELSE without brace
(000001)
          if (a <= 50)                                    // IF with brace
          {
********     cout << "\nNumber is less than 50";
          }                                               /* End of IF with brace */
          else if ((a >= FIFTY)
                   && (a <= 100))                         /* ELSE IF with brace */
          {
********     cout << "\nNumber is between 50 and 100";
          }                                               /* End of ELSE IF with brace */
          else                                            // ELSE with brace
```

```
Out of   16 Intervals,     12 ( 75.00% ) were Executed
       288 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.06.23.00
                                 to 2011-07-22 10.06.38.00
```

CC ANALYZER für C++ - Report for Test Number: 2

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

```
            if (a <= 10)                                        // IF without brace
********       cout << "Number is less than 10";
*       *                                                       // End of IF without brace
            else if ((a >= 10)
                     && (a <= 100))                             // ELSE IF without brace
(000001)       cout << "Number is between 10 and 100";
                                                                // End of ELSE IF without brace
            else                                                // ELSE without brace
********       cout << "Number is greater than 100";
*       *                                                       // ELSE without brace
(000001)
            if (a <= 50)                                        // IF with brace
            {
********       cout << "\nNumber is less than 50";
            }                                                   /* End of IF with brace */
            else if ((a >= FIFTY)
                     && (a <= 100))                             /* ELSE IF with brace */
            {
(000001)       cout << "\nNumber is between 50 and 100";
            }                                                   /* End of ELSE IF with brace */
            else                                                // ELSE with brace
```

```
Out of   16 Intervals,     12 ( 75.00% ) were Executed
       288 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.06.42.00
                                to 2011-07-22 10.06.55.00
```

Accumulated Results of Test Runs

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

```
            if (a <= 10)                                        // IF without brace
********       cout << "Number is less than 10";
*       *                                                       // End of IF without brace
            else if ((a >= 10)
                     && (a <= 100))                             // ELSE IF without brace
(000001)       cout << "Number is between 10 and 100";
                                                                // End of ELSE IF without brace
            else                                                // ELSE without brace
(000001)       cout << "Number is greater than 100";
                                                                // ELSE without brace
(000002)
            if (a <= 50)                                        // IF with brace
            {
********       cout << "\nNumber is less than 50";
            }                                                   /* End of IF with brace */
            else if ((a >= FIFTY)
                     && (a <= 100))                             /* ELSE IF with brace */
            {
(000001)       cout << "\nNumber is between 50 and 100";
            }                                                   /* End of ELSE IF with brace */
            else                                                // ELSE with brace
```

```
Out of   16 Intervals,     12 ( 75.00% ) were Executed
       288 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.06.42.00
                                to 2011-07-22 10.06.55.00
```

CC ANALYZER für Java – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-06-30 16:16:15

Display Summary

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-06-30-15.06.36.890000 / 2011-06-30-15.06.37.380000	6	2	33.00 %
Test 2	2011-06-30-15.07.12.750000 / 2011-06-30-15.07.13.140000	6	5	83.00 %
TOTAL	2011-06-30-15.06.36.890000 / 2011-06-30-15.07.13.140000	6	6	100.00 %

CC ANALYZER für Java - Report for Test Number: 1

SOURCE INPUT FILE: count.java
STATISTICS FILE: statfile_java.dat

```
         import java.io.*;
         public class Count {
            public static void CountChars(Reader in) throws IOException {
********       int count = 0;
*      *       while (in.read() != -1)
********          count++;
*      *
********       System.out.println("No. of chars counted = " + count);
*      *
            }
            public static void main(String[] args) throws Exception {
(000001)       if (args.length >= 1)
********          CountChars(new FileReader(args[0]));
*      *       else
*      *
(000001)          System.err.println("Enter the file name as arg to count chars");
            }
         }
```

```
OUT OF    6 Intervals,     2 ( 33%) WERE EXECUTED
         66 Record(s) Read from Source File
     1 Test Run(s) Represented from 2011-06-30-15.06.36.890000
                                 to 2011-06-30-15.06.37.380000
```

CC ANALYZER für Java - Report for Test Number: 2

SOURCE INPUT FILE: count.java
STATISTICS FILE: statfile_java.dat

```
         import java.io.*;
         public class Count {
            public static void CountChars(Reader in) throws IOException {
(000001)       int count = 0;
               while (in.read() != -1)
(000447)          count++;
(000001)       System.out.println("No. of chars counted = " + count);
            }
            public static void main(String[] args) throws Exception {
(000001)       if (args.length >= 1)
(000001)          CountChars(new FileReader(args[0]));
               else
********          System.err.println("Enter the file name as arg to count chars");
*      *
            }
         }
```

```
OUT OF    6 Intervals,     5 ( 83%) were executed
         66 Record(s) read from Source File
     1 Test Run(s) represented from 2011-06-30-15.07.12.750000
                                 to 2011-06-30-15.07.13.140000
```

Accumulated Results of Test Runs

SOURCE INPUT FILE: count.java
STATISTICS FILE: statfile_java.dat

```
         import java.io.*;
         public class Count {
            public static void CountChars(Reader in) throws IOException {
(000001)       int count = 0;
               while (in.read() != -1)
(000447)          count++;
(000001)       System.out.println("No. of chars counted = " + count);
            }
            public static void main(String[] args) throws Exception {
(000002)       if (args.length >= 1)
(000001)          CountChars(new FileReader(args[0]));
               else
(000001)          System.err.println("Enter the file name as arg to count chars");
            }
         }
```

```
OUT OF    6 Intervals,     6 (100%) were executed
         66 Record(s) read from Source File
     2 Test Run(s) represented from 2011-06-30-15.06.36.890000
                                 to 2011-06-30-15.07.13.140000
```

CC ANALYZER für JSP - HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-07-22 09:15:37

Display Summary

Source Input File: numguess.jsp
Statistics File: statfile_jsp.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-07-22 09.12.55.40 / 2011-07-22 09.12.55.45	4	2	50.00 %
Test 2	2011-07-22 09.12.55.40 / 2011-07-22 09.13.29.25	4	3	75.00 %
TOTAL	2011-07-22 09.12.55.40 / 2011-07-22 09.13.29.25	4	4	100.00 %

CC ANALYZER für JSP - Report for Test Number: 1

Source Input File: numguess.jsp
Statistics File: statfile_jsp.dat

```
           if (numguess.getSuccess()) {
********
                                        %>
           Congratulations!  You got it.
           And after just <%= numguess.getNumGuesses() %> tries.<p>
           <%
********      numguess.reset();
                                    %>
           Care to <a href="numguess.jsp">try again</a>?
         <%
           } else if (numguess.getNumGuesses() == 0) {
(000001)
                                                          %>
           Welcome to the Number Guess game.<p>

           I'm thinking of a number between 1 and 100.<p>

           <form method=get>
           What's your guess? <input type=text name=guess>
           <input type=submit value="Submit">
           </form>
         <%
           } else {
(000002)
```

```
Out of    4 Intervals,      2 ( 50.00% ) were Executed
        105 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 09.12.55.40
                                 to 2011-07-22 09.12.55.45
```

```
                ANALYZER für JSP - Report for Test Number:    2

Source Input File:    numguess.jsp
Statistics File:      statfile_jsp.dat
-------------------------------------------------------------------------------
             if (numguess.getSuccess()) {
(000001)
                                          %>
          Congratulations!  You got it.
          And after just <%= numguess.getNumGuesses() %> tries.<p>
          <%
(000001)     numguess.reset();
                                       %>
          Care to <a href="numguess.jsp">try again</a>?
        <%
           } else if (numguess.getNumGuesses() == 0) {
********
                                                          %>
          Welcome to the Number Guess game.<p>

          I'm thinking of a number between 1 and 100.<p>
          <form method=get>
          What's your guess? <input type=text name=guess>
          <input type=submit value="Submit">
          </form>
        <%
           } else {
(000009)

-------------------------------------------------------------------------------
Out of     4 Intervals,      3 ( 75.00% ) were Executed
         105 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 09.12.55.40
                                 to 2011-07-22 09.13.29.25
```

```
                Accumulated Results of Test Runs

Source Input File:    numguess.jsp
Statistics File:      statfile_jsp.dat
-------------------------------------------------------------------------------
             if (numguess.getSuccess()) {
(000001)
                                          %>
          Congratulations!  You got it.
          And after just <%= numguess.getNumGuesses() %> tries.<p>
          <%
(000001)     numguess.reset();
                                       %>
          Care to <a href="numguess.jsp">try again</a>?
        <%
           } else if (numguess.getNumGuesses() == 0) {
(000001)
                                                          %>
          Welcome to the Number Guess game.<p>
          I'm thinking of a number between 1 and 100.<p>
          <form method=get>
          What's your guess? <input type=text name=guess>
          <input type=submit value="Submit">
          </form>
        <%
           } else {
(000011)

-------------------------------------------------------------------------------
Out of     4 Intervals,      4 ( 100.00% ) were Executed
         105 Line(s) Read from Source File
           4 Record(s) Read from Statistics File
           2 Statistics Record(s) Pertain to This Program
     2 Test Run(s) Represented from 2011-07-22 09.12.55.40
                                 to 2011-07-22 09.13.29.25
```

CC ANALYZER für PL/I Output Results
(C) CC GmbH 2011

```
                    CC ANALYZER für PL/I - STATISTICS-REPORT

LICENSED BY:  CASE CONSULT DEVELOPMENT


  (C) CC GmbH 2011                                  VERSION: 5.4
  DATE: 2011-06-30  16:12:47


  PROGRAM:  DEMOP2

               RUN DATE         INTERVALS EXECUTED PERCENT INSTRUMENTATION DATE
--------------------- --------- -------- ------- --------------------
         2011-06-30(15:03:35)         4        3      75%   2011-06-30(14:26:09)
         2011-06-30(15:07:23)         4        2      50%   2011-06-30(14:26:09)


-------------------------------- REPORT TOTALS --------------------------------

  PROGRAM    RUNS         RUN DATE          INTERVALS   EXECUTED   PERCENT
  --------  -----  ----------------------  ---------  ---------  ---------
  DEMOP2        2         2011-06-30              4          4       100%
  --------  -----                           ---------  ---------  ---------
  TOTAL:        2                                 4          4       100%

101 - REPORT COMPLETED
```

```
          CC ANALYZER für PL/I - DISPLAY-REPORT FOR TEST NUMBER:    1

SOURCE INPUT FILE:   DEMOP2
STATISTICS FILE:     STATFILE


          DEMOP2: PROCEDURE(PARM1,PARM2) REORDER;
(000001)
  NON-PROCEDURAL CODE SUPPRESSED
          PARM1 = 'DEMOP1';
          MAIN_ENTRY_SW = '1'B;
          DISPLAY('PARM2');
          IF PARM2 = 9999 THEN DO;
(000001)      DISPLAY('PARM2 HAS AN IMPOSSIBLE VALUE');
              GOTO PARA_EXIT;
              END;
********  RETURN;
          PARA_EXIT:
(000001)  DISPLAY('ABORTING PROGRAM');
          RETURN;
          END DEMOP2;

OUT OF    4 INTERVALS,     3 ( 75%) WERE EXECUTED.
         59 RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(15:03:35.17)
                                     TO 2011-06-30(15:03:35.25)
```

```
          CC ANALYZER für PL/I - DISPLAY-REPORT FOR TEST NUMBER:    2

SOURCE INPUT FILE:   DEMOP2
STATISTICS FILE:     STATFILE


          DEMOP2: PROCEDURE(PARM1,PARM2) REORDER;
(000010)
  NON-PROCEDURAL CODE SUPPRESSED
          PARM1 = 'DEMOP1';
          MAIN_ENTRY_SW = '1'B;
          DISPLAY('PARM2');
          IF PARM2 = 9999 THEN DO;
********      DISPLAY('PARM2 HAS AN IMPOSSIBLE VALUE');
*      *      GOTO PARA_EXIT;
*      *      END;
(000010)  RETURN;
          PARA_EXIT:
********  DISPLAY('ABORTING PROGRAM');
*      *  RETURN;
*      *  END DEMOP2;

OUT OF    4 INTERVALS,     2 ( 50%) WERE EXECUTED.
         59 RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(15:07:23.62)
                                     TO 2011-06-30(15:07:23.78)
```

```
            CC ANALYZER für PL/I - ACCUMULATED RESULTS OF TEST RUNS

SOURCE INPUT FILE:   DEMOP2
STATISTICS FILE:     STATFILE


          DEMOP2: PROCEDURE(PARM1,PARM2) REORDER;
(000011)
  NON-PROCEDURAL CODE SUPPRESSED
          PARM1 = 'DEMOP1';
          MAIN_ENTRY_SW = '1'B;
          DISPLAY('PARM2');
          IF PARM2 = 9999 THEN DO;
(000001)      DISPLAY('PARM2 HAS AN IMPOSSIBLE VALUE');
              GOTO PARA_EXIT;
              END;
(000011)  RETURN;
          PARA_EXIT:
(000001)  DISPLAY('ABORTING PROGRAM');
          RETURN;
          END DEMOP2;

OUT OF    4 INTERVALS,     4 (100%) WERE EXECUTED.
         59 RECORDS READ FROM SOURCE FILE
          2 TEST RUN(S) REPRESENTED FROM 2011-06-30(15:03:35.17)
                                     TO 2011-06-30(15:07:23.78)
```

CC Deutschland
CC GmbH
Kreuzberger Ring 36
65205 Wiesbaden
Telefon 0611/942040
info-europe@cc-gmbh.de

CC USA
Case Consult Corporation
18 Lyman Street, Suite O
Westborough, MA 01581
Telefon +1-508-651-9898
info-usa@cc-gmbh.de

www.cc-gmbh.de