Tests Unitaires

Introduction

- Dans le paysage dynamique du développement logiciel, la qualité du code revêt une importance cruciale. Les tests unitaires sont l'une des pratiques fondamentales pour garantir la fiabilité, la robustesse et la maintenabilité du code.

- Bien que traditionnellement associés à des langages modernes comme Java, Python ou JavaScript, les tests unitaires jouent également un rôle essentiel dans le développement de logiciels écrits dans des langages plus anciens et pourtant toujours répandus, tels que le COBOL.

- Le COBOL, acronyme de Common Business-Oriented Language, est un langage de programmation impératif largement utilisé dans les applications d'entreprise depuis plusieurs décennies. Malgré sa longue histoire, le COBOL reste au cœur de nombreuses infrastructures informatiques critiques à travers le monde

- Ce cours sur les tests unitaires en COBOL vise à démystifier et à démontrer l'importance des tests unitaires dans le contexte de ce langage.

Exemple concret : Sous-programme "ASSEQ"

```
IDENTIFICATION DIVISION.
PROGRAM-ID. ASSEQ.

DATA DIVISION.
WORKING-STORAGE SECTION.

LINKAGE SECTION.
COPY TESTCONT.
01 TEST-NAME   PIC X(30).
01 EXPECTED    PIC S9(3)V99.
01 ACTUAL      PIC S9(3)V99.

PROCEDURE DIVISION USING TEST-CONTEXT,
                         TEST-NAME,
                               EXPECTED, ACTUAL.

    ADD 1 TO TESTS-RUN

    IF ACTUAL = EXPECTED THEN
          ADD 1 TO PASSES
    ELSE
          DISPLAY 'FAILED : ' TEST-NAME
          DISPLAY 'EXPECTED ' EXPECTED
          DISPLAY 'ACTUAL : ' ACTUAL

          ADD 1 TO FAILURES
    END-IF

    GOBACK.
```

- Ce programme COBOL est un exemple de routine de test unitaire.
- Il est conçu pour être utilisé dans le cadre d'un processus de développement logiciel afin de tester des portions spécifiques de code COBOL.

APIx.SOURCE.COPY(TESTCONT)

```
01  TEST-CONTEXT.
          02  TESTS-RUN      PIC 9(2).
          02  PASSES          PIC 9(2).
          02  FAILURES        PIC 9(2).
```

Exemple concret

- L'intérêt de ce programme réside dans sa capacité à automatiser les tests de petites unités de code COBOL de manière cohérente et reproductible. Il peut être utilisé dans les scénarios suivants :
  - **Tests Unitaires :** Pour vérifier le bon fonctionnement d'une fonction ou d'un module isolé dans un programme COBOL. Par exemple, pour vérifier une fonction de calcul de montant ou de traitement de données.
  - **Tests d'Intégration :** Pour tester l'interaction entre plusieurs modules ou composants dans un programme COBOL. Par exemple, pour vérifier la communication entre un module de gestion des commandes et un module de gestion des stocks.
  - **Tests de Régression :** Pour s'assurer qu'une modification du code n'a pas introduit de nouveaux bugs ou affecté le fonctionnement des fonctionnalités existantes.
- En résumé, ce programme COBOL facilite l'automatisation et la gestion des tests unitaires dans le processus de développement logiciel, ce qui contribue à améliorer la qualité et la fiabilité des applications COBOL.

JCL de compilation "JASSEQ"

```
//JASSEQ   JOB (ACCT#),'STEEVE',MSGCLASS=H,REGION=4M,
//             CLASS=A,MSGLEVEL=(1,1),NOTIFY=&SYSUID,COND=(4,LT)
//*
//*  ETAPE DE COMPILATION
//*
//COMPIL   EXEC IGYWCL,PARM.COBOL=(NODYNAM,ADV,OBJECT,LIB,TEST,APOST)
//COBOL.SYSIN  DD DSN=APIx.SOURCE.COBOL(ASSEQ),DISP=SHR
//COBOL.SYSLIB DD DSN=CEE.SCEESAMP,DISP=SHR
//                DD DSN=APIx.SOURCE.COPY,DISP=SHR
//*
//*  ETAPE DE LINKEDIT
//*
//LKED.SYSLMOD DD DSN=APIx.COBOL.LOAD,DISP=(SHR,KEEP,KEEP)
//LKED.SYSIN       DD *
 NAME ASSEQ(R)
/*
```

Exemple Concret
Programme "SOM"

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SOM.

DATA DIVISION.
WORKING-STORAGE SECTION.
01 WS-RESULTAT PIC S9(3)V99 VALUE 0.

LINKAGE SECTION.
01  INPUT-NUM1                 PIC S9(2)V9(2).
01  INPUT-NUM2                 PIC S9(2)V9(2).
01  RESULT                     PIC S9(3)V9(2).

PROCEDURE DIVISION USING INPUT-NUM1, INPUT-NUM2, RESULT.

    MOVE 0 TO WS-RESULTAT

    COMPUTE WS-RESULTAT = INPUT-NUM1 + INPUT-NUM2
    ON SIZE ERROR
       DISPLAY "-- PBME DE TAILLE --"
    END-COMPUTE

    MOVE WS-RESULTAT TO RESULT
    GOBACK.
```

- Ce programme COBOL est un simple programme qui effectue l'addition de deux nombres décimaux et stocke le résultat dans une variable de résultat.

- Remarques
  - Le programme affiche un message d'erreur si la somme des deux nombres dépasse la capacité de WS-RESULTAT.
  - La variable RESULT est limitée à 4 chiffres avant la virgule et 2 chiffres après la virgule, donc des valeurs plus grandes ou plus précises peuvent entraîner des erreurs de taille

JCL de compilation "JSOM"

```
//JSOM      JOB (ACCT#),'STEEVE',MSGCLASS=H,REGION=4M,
//    CLASS=A,MSGLEVEL=(1,1),NOTIFY=&SYSUID,COND=(4,LT)
//*
//*  ETAPE DE COMPILATION
//*
//COMPIL   EXEC IGYWCL,PARM.COBOL=(NODYNAM,ADV,OBJECT,LIB,TEST,APOST)
//COBOL.SYSIN  DD DSN=APIx.SOURCE.COBOL(SOM),DISP=SHR
//COBOL.SYSLIB DD DSN=CEE.SCEESAMP,DISP=SHR
//                  DD DSN=APIx.SOURCE.COPY,DISP=SHR
//*
//*  ETAPE DE LINKEDIT
//*
//LKED.SYSLMOD DD DSN=APIx.COBOL.LOAD,DISP=(SHR,KEEP,KEEP)
//LKED.SYSIN DD *
 NAME SOM(R)
/*
```

Programme "SOMTEST" Logique de Tests

- Le programme "SOMTEST" est conçu pour tester les capacités de calcul du programme SOM en exécutant plusieurs cas de test prédéfinis.
- Chaque cas de test vérifie si SOM produit les résultats attendus pour différentes paires de nombres d'entrée.
- Le programme exécute cinq cas de test différents :
  - **SOM-OF-TWO-POS-TEST** : Teste l'addition de deux nombres positifs.
  - **SOM-OF-ONE-NEG-TEST** :Teste l'addition d'un nombre négatif et d'un nombre positif.
  - **SOM-OF-TWO-NEG-TEST** :Teste l'addition de deux nombres négatifs.

Steeve ASSOUS
pour

Programme Complet "SOMTEST"

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SOMTEST.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    DECIMAL-POINT IS COMMA.

DATA DIVISION.
WORKING-STORAGE SECTION.

01  INPUT-NUMBER1      PIC S9(2)V9(2).
01  INPUT-NUMBER2      PIC S9(2)V9(2).

01  RESULT             PIC S9(3)V9(2).
01  EXPECTED-RESULT    PIC S9(3)V9(2).
01  LIB PIC X(30).
01  L-SEP PIC X(30) VALUE ALL "*".

LINKAGE SECTION.
COPY TESTCONT.
```

```
PROCEDURE DIVISION USING TEST-CONTEXT.

    PERFORM SOM-OF-TWO-POS-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-ONE-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-TWO-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    GOBACK.

SOM-OF-TWO-POS-TEST.
    MOVE 0 TO RESULT
    MOVE 5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 12,45 TO EXPECTED-RESULT
    MOVE 'SOM-OF-TWO-POS-TEST' TO LIB
    CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
    DISPLAY "******** ", LIB, "*********"
    CALL 'ASSEQ' USING TEST-CONTEXT, LIB, EXPECTED-RESULT,
RESULT.
```

```
 SOM-OF-ONE-NEG-TEST.
    MOVE 0 TO RESULT
    MOVE -5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 2,25 TO EXPECTED-RESULT

    MOVE 'SOM-OF-ONE-NEG-TEST' TO LIB
    CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
    DISPLAY "******** ", LIB, "*********"

    CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
                       EXPECTED-RESULT, RESULT.

 SOM-OF-TWO-NEG-TEST.
     MOVE 0 TO RESULT
     MOVE -5,84  TO INPUT-NUMBER1
     MOVE -7,29  TO INPUT-NUMBER2
     MOVE -13,13 TO EXPECTED-RESULT

     MOVE 'SOM-OF-TWO-NEG-TEST' TO LIB
     CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
     DISPLAY "******** ",
     CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
                        EXPECTED-RESULT, RESULT.
```

Steeve ASSOUS
pour

Programme "SOMTEST"
Processus de Test

- Pour chaque cas de test, le programme :
  - Initialise les variables d'entrée et le résultat attendu.
  - Appelle le programme SOM pour effectuer l'addition.
  - Affiche le nom du test.
  - Utilise un programme auxiliaire ASSEQ pour comparer le résultat obtenu avec le résultat attendu.
  - Affiche le nombre de tests réussis et échoués.

```
SOM-OF-TWO-POS-TEST.

    MOVE 0 TO RESULT
    MOVE 5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 12,45 TO EXPECTED-RESULT
    MOVE 'SOM-OF-TWO-POS-TEST' TO LIB

    CALL 'SOM' USING INPUT-NUMBER1, INPUT-
NUMBER2, RESULT

    DISPLAY "******** ", LIB, "**********"

    CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
EXPECTED-RESULT, RESULT.
```

Steeve ASSOUS
pour

Programme Complet "SOMTEST"

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SOMTEST.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    DECIMAL-POINT IS COMMA.

DATA DIVISION.
WORKING-STORAGE SECTION.

01  INPUT-NUMBER1      PIC S9(2)V9(2).
01  INPUT-NUMBER2      PIC S9(2)V9(2).

01  RESULT             PIC S9(3)V9(2).
01  EXPECTED-RESULT    PIC S9(3)V9(2).
01  LIB PIC X(30).
01  L-SEP PIC X(30) VALUE ALL "*".

LINKAGE SECTION.
COPY TESTCONT.
```

```
PROCEDURE DIVISION USING TEST-CONTEXT.

    PERFORM SOM-OF-TWO-POS-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-ONE-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-TWO-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    GOBACK.

SOM-OF-TWO-POS-TEST.
    MOVE 0 TO RESULT
    MOVE 5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 12,45 TO EXPECTED-RESULT
    MOVE 'SOM-OF-TWO-POS-TEST' TO LIB
    CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
    DISPLAY "******** ", LIB, "*********"
    CALL 'ASSEQ' USING TEST-CONTEXT, LIB, EXPECTED-RESULT,
RESULT.
```

```
 SOM-OF-ONE-NEG-TEST.
    MOVE 0 TO RESULT
    MOVE -5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 2,25 TO EXPECTED-RESULT

    MOVE 'SOM-OF-ONE-NEG-TEST' TO LIB
    CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
    DISPLAY "******** ", LIB, "*********"

    CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
                       EXPECTED-RESULT, RESULT.

 SOM-OF-TWO-NEG-TEST.
     MOVE 0 TO RESULT
     MOVE -5,84  TO INPUT-NUMBER1
     MOVE -7,29  TO INPUT-NUMBER2
     MOVE -13,13 TO EXPECTED-RESULT

     MOVE 'SOM-OF-TWO-NEG-TEST' TO LIB
     CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
     DISPLAY "******** ",
     CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
                        EXPECTED-RESULT, RESULT.
```

Programme "SOMTEST"
Résultats des Tests

- Après chaque cas de test, le programme affiche les résultats cumulés des tests exécutés, indiquant le nombre de tests réussis (OK) et échoués (KO).

- Une ligne de séparation est affichée entre chaque série de résultats pour une meilleure lisibilité.

```
PROCEDURE DIVISION USING TEST-CONTEXT.

    PERFORM SOM-OF-TWO-POS-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-ONE-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-TWO-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    GOBACK.
```

Steeve ASSOUS
pour

Programme Complet "SOMTEST"

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SOMTEST.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    DECIMAL-POINT IS COMMA.

DATA DIVISION.
WORKING-STORAGE SECTION.

01  INPUT-NUMBER1      PIC S9(2)V9(2).
01  INPUT-NUMBER2      PIC S9(2)V9(2).

01  RESULT             PIC S9(3)V9(2).
01  EXPECTED-RESULT    PIC S9(3)V9(2).
01  LIB PIC X(30).
01  L-SEP PIC X(30) VALUE ALL "*".

LINKAGE SECTION.
COPY TESTCONT.
```

```
PROCEDURE DIVISION USING TEST-CONTEXT.

    PERFORM SOM-OF-TWO-POS-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-ONE-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    PERFORM SOM-OF-TWO-NEG-TEST.
    DISPLAY 'RUN ', TESTS-RUN, ' OK ', PASSES, ' KO ', FAILURES
    DISPLAY L-SEP

    GOBACK.

SOM-OF-TWO-POS-TEST.
    MOVE 0 TO RESULT
    MOVE 5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 12,45 TO EXPECTED-RESULT
    MOVE 'SOM-OF-TWO-POS-TEST' TO LIB
    CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
    DISPLAY "******** ", LIB, "*********"
    CALL 'ASSEQ' USING TEST-CONTEXT, LIB, EXPECTED-RESULT,
RESULT.
```

```
 SOM-OF-ONE-NEG-TEST.
    MOVE 0 TO RESULT
    MOVE -5,10  TO INPUT-NUMBER1
    MOVE 7,35  TO INPUT-NUMBER2
    MOVE 2,25 TO EXPECTED-RESULT

    MOVE 'SOM-OF-ONE-NEG-TEST' TO LIB
    CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
    DISPLAY "******** ", LIB, "*********"

    CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
                       EXPECTED-RESULT, RESULT.

 SOM-OF-TWO-NEG-TEST.
     MOVE 0 TO RESULT
     MOVE -5,84  TO INPUT-NUMBER1
     MOVE -7,29  TO INPUT-NUMBER2
     MOVE -13,13 TO EXPECTED-RESULT

     MOVE 'SOM-OF-TWO-NEG-TEST' TO LIB
     CALL 'SOM' USING INPUT-NUMBER1, INPUT-NUMBER2,
RESULT
     DISPLAY "******** ",
     CALL 'ASSEQ' USING TEST-CONTEXT, LIB,
                        EXPECTED-RESULT, RESULT.
```

Steeve ASSOUS
pour

JCL de Compilation "JSOMTEST"

```
//JSOMTEST JOB (ACCT#),'STEEVE',MSGCLASS=H,REGION=4M,
//    CLASS=A,MSGLEVEL=(1,1),NOTIFY=&SYSUID,COND=(4,LT)
//******************  ETAPE DE COMPILATION ********************
//COMPIL   EXEC IGYWCL,PARM.COBOL=(ADV,OBJECT,LIB,TEST,APOST)
//COBOL.SYSIN  DD DSN=APIx.SOURCE.COBOL(SOMTEST),DISP=SHR
//COBOL.SYSLIB DD DSN=CEE.SCEESAMP,DISP=SHR
//                    DD DSN=APIx.SOURCE.COPY,DISP=SHR
//******************  ETAPE DE LINKEDIT ***********************
//LKED.SYSLMOD DD DSN=APIx.COBOL.LOAD,DISP=(SHR,KEEP,KEEP)
//LKED.SYSLIB     DD DSN=APIx.COBOL.LOAD,DISP=SHR
//LKED.SYSIN DD *
  INCLUDE SYSLIB('ASSEQ')
  INCLUDE SYSLIB('SOM')
  NAME SOMTEST(R)
/*
```

JCL d'Exécution "JTESTS"

```
//JTESTS    JOB (ACCT#),'STEEVE',MSGCLASS=H,REGION=4M,
//    CLASS=A,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*
//*  ETAPE EXECUTION
//*
//JSOMT    EXEC PGM=SOMTEST
//STEPLIB  DD DSN=APIx.COBOL.LOAD,DISP=SHR
//SYSOUT   DD SYSOUT=*
```

Conclusion

- Ce programme est essentiellement un script de test automatisé qui vérifie la précision et la robustesse du programme SOM en testant diverses conditions et en validant les résultats obtenus par rapport aux résultats attendus.