Unit Test

소프트웨어공학개론

201311306 이진호

201511266 배율희

20123189 박성규

Unit Test

소프트웨어공학개론

1. UNIT TEST 개념

   프로그램의 특정 모듈이 정확하게 동작하는지를 검증하는 절차이다.
   즉, 모든 함수와 메소드에 대한 테스트 케이스를 작성하는 절차를 말한다.
   프로그램을 작은 단위로 쪼개서 검사하기에 문제점의 발견이 용이하며, 프로그래머가 다시 코드를 만들더라도, 해당 모듈이 의도한대로 움직이는지를 쉽게 알 수 있어서 변경이 쉽다.
   또한 하위 단위를 합쳐서 상위 단위를 만드는 방식이기에 통합 테스트에서 더욱 더 빛을 발한다.

2. UNIT TEST 수행방법

   1) 프로그램을 작은 단위로 쪼갠다.
   2) 각 단위에 대한 단위테스트를 만든다.
   3) 단위테스트는 첫째, 발생할 수 있는 경우에 대해 올바른 결과가 나오는가, 둘째, 오류의 발생에 대해 적절하게 예외처리가 되는가 등을 고려한 단위테스트를 작성한다.
   4) 이를 수행한 뒤에, 만약에 결과가 잘못되었다면 디버깅 및 수정한다.
   5) 이 과정을 반복해서 원하는 결과를 얻는다.

3. UNIT TEST 를 위한 도구

   테스트를 하기 위한 프로그램을 기초부터 만들기에는 힘이 들고, 만약에 그런 프로그램을 만들더라도 신뢰성이 보장되기 힘들기 때문에, 신뢰성이 보장된 단위테스트를 수행하기 위해서 도구를 사용한다.

1) Cunit

-KISS 원칙에 따른다. 작고 간단하고 알기 쉽게 만들어진 유닛테스트 시스템이다. C 언어 기반의 프레임워크이다.

-C 언어로 쓰고, 수정하고, 실행하는 시스템이다.

정적 라이브러리와 연결되어 있다.

-내부에 내장된 함수들이 테스트 과정을 처리해준다.

2) Unity

-Unity 는 100% 순수 C 언어로 작성되었다.

-대부분의 임베디드 컴파일러를 지원하는 ANSI 표준을 따른다.

-8bit 부터 64bit 까지 동일한 테스트 실행 방법을 제공한다.

3) Check

-Check 는 유닛테스트 정의를 위해 간단한 인터페이스를 제공한다.

-테스트는 별도의 어드레스에서 실행된다.

-세그멘테이션 오류, 시그널 캐치와 같은 코드에러와 장애를 확인 가능하다.

*Check test code 작성법

(모듈 이름).c;

```
#include <check.h>

START_TEST(테스트 이름 1)
{
    테스트 내용
}
END_TEST

...

START_TEST(테스트 이름 N){ 테스트 내용}END_TEST




Suite *
my_suite(void)
{
 Suite *s = suite_create(Suite 이름);
 TCase *tc = tcase_create(TCase 이름);
 tcase_add_test(tc, 테스트 이름 1);
    ...
 tcase_add_test(tc, 테스트 이름 N);
 suite_add_tcase(s, tc);
 return s;
}

int main(void)
{
    Suite *s = my_suite();
    SRunner *sr = srunner_create(s);
    srunner_run_all(sr, CK_ENV);
    return 0;
}
```

cygwin

$ CK_VERBOSITY = ("silent"/"minimal"/"normal"/"verbose" 중 하나) ./(모듈 이름)

*참조 : http://egloos.zum.com/studyfoss/v/5346826

4. 실행

*청소 State 구현(간략화)

```
#include <stdio.h>

void clean(int* state1, int* water);

typedef struct{
  int water;//물 잔 량
  int beans;//원 두 잔 량
  int coffeePowder;//커 피 가 루 잔 량
//이 하 생 략

}currentStatus;

int main(){

  currentStatus s = {1000, 1000, 1000};//임 의 로  만 듬  현 재  물  원 두  가 루  잔 량
  int currentState = 0;//0이  대 기 중  그 외 값 은  청 소 중  가 열 중  등 등

  clean(&currentState, &(s.water));


  return 0;
}

void clean(int* state1, int* water){ //state1은

  if(*state1==0){//0이  대 기 중 인 상 태 를  나 타 낸 다 고  가 정


    if(*water < 500 ){
      printf("물  잔 량 이  부 족 합 니 다 .");

    }
    else{
      printf("청 소 를  진 행 합 니 다 .");
      *water += 500;
      *state1 = 1;
    }

  }
  else{
    printf("대 기 중 이  아 닙 니 다 .");

  }

}
~
~
~
                                                    16,3          모 두
```

*유닛테스트 결과

~/sogongCUnit/CUnit-2.1-2

```
#include <stdio.h>
#include <CUnit/CUnit.h>
#include <CUnit/Basic.h>

int clean(int* state1, int* water);
void test_clean();
static int InitSuite1();
static int CleanupSuite1();

typedef struct{
  int water;//물 잔 량
  int beans;//원 두 잔 량
  int coffeePowder;//커 피 가 루 잔 량
//이 하 생 략

}currentStatus;

CU_TestInfo test_array1[]={
  {"test1",clean},
  CU_TEST_INFO_NULL
};

CU_SuiteInfo suite_array[]={
  {"suite1", InitSuite1, CleanupSuite1,test_array1},
  CU_SUITE_INFO_NULL

};

static int InitSuite1(){
  return 0;
}

static int CleanupSuite1(){
  return 0;
}

int main(){

  //currentStatus s = {1000, 1000, 1000};//임 의 로  만 듬  현 재  물  원 두  가 루  잔 량
  //int currentState = 0;//0이  대 기 중  그 외 값 은  청 소 중  가 열 중  등 등

  //clean(&currentState, &(s.water));
  if (CUE_SUCCESS != CU_initialize_registry())
        return CU_get_error();
  CU_register_suites(suite_array);

  CU_basic_run_tests();

  test_clean();

  CU_cleanup_registry();

  return 0;
}

int clean(int* state1, int* water){ //state1은

  if(*state1==0){//0이  대 기 중 인 상 태 를  나 타 낸 다 고  가 정


    if(*water < 500 ){
                                                              1,1           꼭 대 기
```


~/sogongCUnit/CUnit-2.1-2
return 0;
}
int main(){
//currentStatus s = {1000, 1000, 1000};//임의로 만듬 현재 물 원두 가루 잔량
//int currentState = 0;//0이 대기중 그외값은 청소중 가열중 등등
//clean(&currentState, &(s.water));
if (CUE_SUCCESS != CU_initialize_registry())
return CU_get_error();
CU_register_suites(suite_array);
CU_basic_run_tests();
test_clean();
CU_cleanup_registry();
return 0;
}
int clean(int* state1, int* water){ //state1은
if(*state1==0){//0이 대기중인상태를 나타낸다고 가정
if(*water < 500){
printf("물 잔량이 부족합니다.");
return -1;
}
else{
printf("청소를 진행합니다.");
*water += 500;
*state1 = 1;
return 0;
}
}
else{
printf("대기중이 아닙니다.");
return -2;
}
}
void test_clean(){
currentStatus s = {1000,1000,1000};
int currentState = 0;
CU_ASSERT(clean(&currentState, &(s.water)) == 0);
currentState = 1;
CU_ASSERT(clean(&currentState, &(s.water)) == -2);
s.water -= 600;
CU_ASSERT(clean(&currentState, &(s.water)) == -1);
}
94,1
바닥

```
clock@DESKTOP-B8K9NCC ~/sogongCUnit/CUnit-2.1-2
$ ./mainTest


     CUnit - A unit testing framework for C - Version 2.1-2
     http://cunit.sourceforge.net/

대 기 중 이  아 닙 니 다 .

Run Summary:    Type  Total    Ran Passed Failed Inactive
              suites      1      1    n/a      0        0
               tests      1      1      1      0        0
             asserts      0      0      0      0      n/a

Elapsed time =    0.000 seconds
청 소 를  진 행 합 니 다 .assertion "NULL != f_pCurSuite" failed: file "TestRun.c", line 159
, function: CU_assertImplementation
Aborted (core dumped)
```