BSM462
Yazılım Testi

Hafta - 7

Google C++ Test Framework

Dr. Öğr. Üyesi M. Fatih ADAK

fatihadak@sakarya.edu.tr

İçerik

- Tanım
- Google C++ Test Yetenekleri
- Test Suite ve Test Case
- Temel Kavramlar
- Test Case
- Test Sınıfı Oluşturma
- Test Öncesi ve Sonrası Metotlar
- Assert Türleri
- Assert ve Expect

Tanım

- Test teknoloji takımı tarafından geliştirilmiştir.
- Google'ın özel gereksinimleri ve kısıtları göz önünde bulundurularak geliştirilen bir test çerçevesidir.
- İşletim sistemi veya derleyici önemli değil her C++ kodu test edilebilir.
- Birim testi ile birlikte birçok testi desteklemektedir.
- Google test xUnit mimarisini temel almaktadır. Mimari olarak JUnit'e çok benzemektedir.

Google C++ Test Yetenekleri

- Testler bağımsız çalıştırılabilir. Bir test başarısız diye diğer testleri etkilemez.
- Google test, her bir testi farklı bir nesne üzerinde çalıştırarak testleri izole eder.
- Google test, birbirleri ile ilgili testleri gruplayabilir. Bu grupta bilgi paylaşımı yapılabilir.
- Böylelikle kararlı bir yapı sunar.
- Testler platform bağımsız olup taşınabilirler.
- Hızlı bir test mekanizması sağlar.

Test Suite ve Test Case Terimleri

- Google Test, birbirleri ile alakalı testleri gruplandırmak için Test Case terimini kullanır.
- Günümüz akademik çalışmalarında ve Uluslararası Yazılım Test Yeterlilikler Kurulu (ISTQB) aynı işlem için Test Suite terimini kullanmaktadırlar.
- Google Test program testi için TEST() terimini kullanırken, ISTQB ise Test Case terimini tercih etmektedir.

Temel Kavramlar

- Google Test kullanıp işe başlamak için bir durumun doğru olup olmadığını kontrol eden assert ifadeleri kullanılmalıdır.
- Bir assert ifadesi 3 farklı sonuç üretebilir
  - Başarılı (success)
  - Önemsiz Başarısızlık (nonfatal failure)
  - Önemli Başarısızlık (fatal failure)
- Eğer öneli başarısızlık oluşmuş ise o anki fonksiyonu sonlandırır.
- Aksi takdirde program normal devam eder.
- Birden fazla test aynı nesneyi veya modülü paylaşımlı kullanacaklarsa bu testler **test fixture** sınıfına konulmalıdırlar.

Test Öncesi ve Sonrası Metotlar

- Tasarlanan bir test sınıfı içindeki her bir test metoduna özel öncesi ve sonrası çalışacak metotlar yazılabilir.
  - virtual void SetUp() { ... } // Her bir test için tanımlanabilecek test öncesi metot
  - virtual void TearDown() { ... }// Her bir test için tanımlanabilecek test sonrası metot

ASSERT Türleri

TEST	FATAL	NON FATAL
True	ASSERT_TRUE(*condition*)	EXPECT_TRUE(*condition*)
False	ASSERT_FALSE(*condition*)	EXPECT_FALSE(*condition*)
Equal	ASSERT_EQ(*arg1*,*arg2*)	EXPECT_EQ(*arg1*,*arg2*)
Not Equal	ASSERT_NE(*arg1*,*arg2*)	EXPECT_NE(*arg1*,*arg2*)
Less Than	ASSERT_LT(*arg1*,*arg2*)	EXPECT_LT(*arg1*,*arg2*)
Less Than or Equal	ASSERT_LE(*arg1*,*arg2*)	EXPECT_LE(*arg1*,*arg2*)
Greater Than	ASSERT_GT(*arg1*,*arg2*)	EXPECT_GT(*arg1*,*arg2*)
Greater Than or Equal	ASSERT_GE(*arg1*,*arg2*)	EXPECT_GE(*arg1*,*arg2*)
C String Equal	ASSERT_STREQ(*str1*,*str2*)	EXPECT_STREQ(*str1*,*str2*)
C String Not Equal	ASSERT_STRNE(*str1*,*str2*)	EXPECT_STRNE(*str1*,*str2*)
C String Case Equal	ASSERT_STRCASEEQ(*str1*,*str2*)	EXPECT_STRCASEEQ(*str1*,*str2*)
C String Case Not Equal	ASSERT_STRCASENE(*str1*,*str2*)	EXPECT_STRCASENE(*str1*,*str2*)
Verify that exception is thrown	ASSERT_THROW(*statement*,*exception_type*)	EXPECT_THROW(*statement*,*exception_type*)
Verify that exception is thrown	ASSERT_ANY_THROW(*statement*)	EXPECT_ANY_THROW(*statement*)
Verify that exception is NOT thrown	ASSERT_NO_THROW(*statement*)	EXPECT_NO_THROW(*statement*)

ASSERT ve EXPECT

- Eğer assert başarısız olursa test işleyişi sona erer.
- Eğer expect başarısız olursa işleyişin devam etmesine izin verir.
- Her ikisinde de sonuç olarak test başarısız olacaktır.

Referanslar

- Naik, Kshirasagar, and Priyadarshi Tripathy. *Software testing and quality assurance: theory and practice*. John Wiley & Sons, 2011.
- Ammann, Paul, and Jeff Offutt. *Introduction to software testing*. Cambridge University Press, 2016.
- Padmini, C. "Beginners Guide To Software Testing." (2004).
- Archer, Clark, and Michael Stinson. *Object-Oriented Software Measures*. No. CMU/SEI-95-TR-002. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1995.
- Pandey, Ajeet Kumar, and Neeraj Kumar Goyal. *Early Software Reliability Prediction*. Springer, India, 2015.
- Koskela, L. (2013). *Effective unit testing: A guide for Java developers*. Manning.