HOGESCHOOL VAN AMSTERDAM

Automated Testing

Project Game Technology

Wouter van der Hauw, 500706687

Maart 2017

Glenn Hoff, Wouter van de Hauw, Michael Fuents Rodriguez, Rosa Corstjens, Guus van Gend, Kevin de Leeuw, Frederick van der Meulen & Jasper Meier

Project Game Technology
Team 5

Under the guidance of
Alexander Mulder, Product Owner
Anders Bouwer, Scrum coach
Eric Klok, Technisch consultant

Automated Testing

Project Game Technology

Wouter van der Hauw, 500706687

Game Technology
Hogeschool van Amsterdam
Maart 2017

Choosen Testing Framework

As a team, we choose the CppUnit test framework. CppUnit is the C++ port of the Junit framework. The Test output of this framework is a XML file for automatic testing. Each unit test begins with the setup() method and followed with the test and then ending with the teardown() method. Each unit test exist of a assert() method to test the outcome of method.

Reason for chosen Testing Framework

The reason why to choose this testing framework is because it is the most widely used unit-testing framework. Also, a reason is because OGRE already uses CppUnit for a selection of unit test that covers the basic. Also, because in CppUnit you need a minimal amount of work to add new tests to your project.

4 examples of applications

In CppUnit you got multiple options to use Assert.

```
class ComplexNumberTest : public CppUnit::TestCase {
public:
  ComplexNumberTest( std::string name ) : CppUnit::TestCase( name ) {}

  void runTest() {
    CPPUNIT_ASSERT( Complex (10, 1) == Complex (10, 1) );
    CPPUNIT_ASSERT( !(Complex (1, 1) == Complex (2, 2)) );
  }
};
```

Example 1

If you look to example 1, you can see that in this example they have used the function CPPUNIT_ASSERT(condition). In this function is an expression passed and if this expression is true then this test is passed.

```
void fractiontest :: addTest (void)
  {
    // check subtraction results
    CPPUNIT_ASSERT_EQUAL (*a + *b, Fraction (7, 6));
    CPPUNIT_ASSERT_EQUAL (*b + *c, Fraction (1));
    CPPUNIT_ASSERT_EQUAL (*d + *e, Fraction (-5));
      ...
```

Example 2

If You look to example 2, you can see that this method uses the function: CPPUNIT_ASSERT_EQUAL(). This function check if the first value is the same as the last value and then the test is passed.

```
void fractiontest :: exceptionTest (void)
{
  // an exception has to be thrown here
  CPPUNIT_ASSERT_THROW (Fraction (1, 0),
          DivisionByZeroException);
}
```

Example 3

In example 3 you can see the function: CPPUNIT_ASSERT_THROW(). In this function, we first run the expression and then check if this throws the same exception as excepted.

```
  std::vector<int> v;
  v.push_back( 10 );
CPPUNIT_ASSERT_NO_THROW( "std::vector<int> v;", v.at( 0 ) );
```

Example 4

In example 4 you can see the function: CPPUNIT_ASSERT_NO_THROW(). In this function you check if some code doesn't throw an exception and if this is true then the test is passed.

REFERENCES

http://www.ogre3d.org/tikiwiki/Visual+Unit+Testing+Framework
https://freedesktop.org/wiki/Software/cppunit/
http://gamesfromwithin.com/exploring-the-c-unit-testing-framework-jungle#cppunit

codesnippet:
http://cppunit.sourceforge.net/doc/cvs/group___assertions.html#ga13
http://www.uow.edu.au/~nabg/222/Lectures/P3UnittestingforC++.pdf