Google Test Framework

Лекция 8

Подключение Google Test Framework

1. Получение исходного кода Framework'а:

```
>svn checkout
http://googletest.googlecode.com/svn/tags/release-1.6.0
```

2. Подключение INCLUDE:

```
Path/gtest/1.6.0/src/
```

3. Подключение исходного кода для компиляции:

```
gtest-all.cc
```

Формирование консольного приложения для тестирования

```
#include "gtest/gtest.h"

int main(int argc, char** argv)
{
	::testing::InitGoogleTest( &argc, argv );
	if ( !RUN_ALL_TESTS() )
		std::cout << "Passed!" << std::endl;
	else
		std::cout << "FAILED!" << std::endl;

	return 0;
}
```

Простейшее тестирование

```
//! Вычисление факториала.
/*!
    \return значение факториала.
    \retval 1 если number = 0 или number = 1.
    \param number считаем число "number!".
*/
unsigned int
factorial( unsigned int number )
{
    if ( number < 2 )
        return 1;

    return number * factorial( number - 1 );
}

TEST( Factorial, Simple )
{
    EXPECT_EQ( 1, factorial( 1 ) );
    EXPECT_EQ( 2, factorial( 2 ) );
    EXPECT_EQ( 6, factorial( 3 ) );
    EXPECT_TRUE( factorial( 20 ) > 1e10 );
}
```

ASSERT и EXPECT

```
TEST( Factorial, Simple )
{
      EXPECT_EQ( 1, factorial( 1 ) );
      ASSERT_EQ( 2, factorial( 2 ) );
      ASSERT_EQ( 6, factorial( 3 ) );
      ASSERT_FALSE( factorial( 20 ) < 1e8 );
}
```

Группы и тесты

```
TEST ( <Имя группы>, <Имя теста> )
{
      [..]
}
```

Выполнение теста как функции

```
TEST( HumidifierMethods, PassTimeIsOff )
{
        humidifier_t first;

        // Проверка что действительно выключен.
        ASSERT_FALSE( first.is_on() );
        unsigned int stored = first.stored();
        // Проверяем, что в воздух ничего не отправилось.
        EXPECT_EQ( 0, first.pass_time( 10 ) );
        // Запас воды не должен изменится.
        ASSERT_EQ( stored, first.stored() );

        // Добавим воды и повторим действия.
        first.add_water( 10 );
        stored = first.stored();
        // Проверяем, что в воздух ничего не отправилось.
        EXPECT_EQ( 0, first.pass_time( 10 ) );
        // Запас воды не должен изменится.
        ASSERT_EQ( stored, first.stored() );
}
```

Fixture-tests

```
class QuickTest : public testing::Test
{
    protected:
        virtual void
        SetUp()
        {
            m_start_time = time(NULL);
        }

        virtual void
        TearDown()
        {
             const time_t end_time = time(NULL);
             EXPECT_TRUE(end_time - m_start_time <= 1) <<
                "The test took too long.";
        }
        time_t m_start_time;
};
```

Использование готовых fixture

```
class IntegerFunctionTest : public QuickTest {
};

TEST_F( IntegerFunctionTest, One )
{
	EXPECT_EQ( 1e9, func( 1e9 ) );
	EXPECT_EQ( 1, func( 1 ) );
}

TEST_F( IntegerFunctionTest, Two )
{
	EXPECT_EQ( 1, func(1) );
	EXPECT_EQ( 2, func(2) );
}
```