Magellan

A Simple xUnit Test Framework in Modern C++11

刘光聪

This page is intentionally left blank.

目录

This page is intentionally left blank.

I'm not a great programmer; I'm just a good programmer with great habits.

- Kent Beck

1 初识 Magellan

1.1 简介

Magellan 是一个简单的、可扩展的、使用 C++11 实现的 xUnit 测试框架，Magellan 设计灵感来自于 Java 社区著名的测试框架 JUnit。

1.2 安装

1.2.1 环境准备

编译环境

Magellan 目前仅在 Linux, MAC OS X 系统上测试过，仅支持 GCC4.8 及以上版本，CLANG 3.4 及以上版本。

安装 CMake

CMake 的下载地址是：http://www.cmake.org。以 Ubuntu 为例，使用 apt-get 安装 CMake。

示例代码 1-1 安装 CMAKE

```
$ sudo apt-get install cmake
```

安装 l0-infra

Magellan 依赖于 l0-infra，所以必须先安装 l0-infra

示例代码 1-2 安装 RVM

```
$ git clone https://gitlab.com/horance/l0-infra.git
$ cd l0-infra
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
```

1.2.2 安装 Magellan

示例代码 1-3 安装 RVM

```
$ git clone https://gitlab.com/horance/magellan.git
$ cd magellan
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
```

1.3 破冰之旅

1.3.1 第一个测试用例

示例代码 1-4 test/quantity/LengthTest.cpp

```
#include <magellan/magellan.hpp>
#include "quantity/length/Length.h"

USING_HAMCREST_NS

FIXTURE(LengthTest)
{
    TEST("1 FEET should equal to 12 INCH")
    {
        ASSERT_THAT(Length(1, FEET), eq(Length(12, INCH)));
    }

    TEST("1 YARD should equal to 3 FEET")
    {
        ASSERT_THAT(Length(1, YARD), eq(Length(3, FEET)));
    }

    TEST("1 MILE should equal to 1760 YARD")
    {
        ASSERT_THAT(Length(1, MILE), eq(Length(1760, YARD)));
    }
};
```

Magellan 使用 Hamcrest 的断言机制，使得断言更加统一、自然，且具有良好的扩展性。

1.3.2 Length 实现

示例代码 1-5 test/quantity/Length.h

```
#include "quantity/base/Amount.h"

enum LengthUnit
{
    INCH = 1,
    FEET = 12 * INCH,
    YARD = 3 * FEET,
    MILE = 1760 * YARD,
};

struct Length
{
    Length(Amount amount, LengthUnit unit);

    bool operator==(const Length&) const;
    bool operator!=(const Length&) const;

private:
    const Amount amountInBaseUnit;
};
```

示例代码 1-6 test/quantity/Length.cpp

```
#include "quantity/base/Length.h"

Length::Length(Amount amount, LengthUnit unit)
  : amountInBaseUnit(unit * amount)
{
}

bool Length::operator==(const Length& rhs) const
{
    return amountInBaseUnit == rhs.amountInBaseUnit;
}

bool Length::operator!=(const Length& rhs) const
{
    return !(*this == rhs);
}
```

1.3.3 Main 函数

示例代码 1-7 test/main.cpp

```
#include "magellan/magellan.hpp"

int main(int argc, char** argv)
{
    return magellan::run_all_tests(argc, argv);
}
```

1.3.4 CMakeLists 构建脚本

示例代码 1-8 quantity/CMakeLists.txt

```
project(quantity)

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++0x")

include_directories(${CMAKE_CURRENT_SOURCE_DIR}/include)

FILE(GLOB_RECURSE all_files
*.cpp
*.cc
*.c++
*.c
*.C)

add_executable(quantity-test ${all_files})
target_link_libraries(quantity-test magellan l0-infra)
```

1.3.5 构建 Quantity

示例代码 1-9 构建 Quantity，并执行测试

```
$ mkdir build
$ cd build
$ cmake ..
$ make
```

1.3.6 执行测试

示例代码 1-10 执行测试

```
$ ./quantity-test

[==========] Running 3 test cases.
[----------] 3 tests from All Tests
[----------] 3 tests from LengthTest
[ RUN      ] LengthTest::1 FEET should equal to 12 INCH
[       OK ] LengthTest::1 FEET should equal to 12 INCH(40 us)
[ RUN      ] LengthTest::1 YARD should equal to 3 FEET
[       OK ] LengthTest::1 YARD should equal to 3 FEET(40 us)
[ RUN      ] LengthTest::1 MILE should equal to 1760 YARD
[       OK ] LengthTest::1 MILE should equal to 1760 YARD(40 us)
[----------] 3 tests from LegnthTest

[----------] 3 tests from All Tests

[==========] 3 test cases ran.
[  TOTAL   ] PASS: 3  FAILURE: 0  ERROR: 0  TIME: 120 us
```

This page is intentionally left blank.

Write programs for people first, computers second.

- Steve McConnell

2 用例设计

2.1 Fixture

2.1.1 Fixture

示例代码 2-1 test/quantity/LengthTest.cpp

```
#include <magellan/magellan.hpp>

FIXTURE(LengthTest)
{
};
```

FIXTURE 的参数可以是任意的 C/C++ 标识符。一般而言，将其命名为 CUT(Class Under Test) 的名字即可。

2.2 Test

2.2.1 自动标识

Magellan 能够自动地实现测试用例的标识功能，用户可以使用字符串来解释说明测试用例的意图，使得用户在描述用例时更加自然和方便。

示例代码 2-2 test/quantity/LengthTest.cpp

```
#include <magellan/magellan.hpp>
#include "quantity/length/Length.h"

USING_HAMCREST_NS

FIXTURE(LengthTest)
{
    TEST("1 FEET should equal to 12 INCH")
    {
        ASSERT_THAT(Length(1, FEET), eq(Length(12, INCH)));
    }

    TEST("1 YARD should equal to 3 FEET")
    {
        ASSERT_THAT(Length(1, YARD), eq(Length(3, FEET)));
    }

    TEST("1 MILE should equal to 1760 YARD")
    {
        ASSERT_THAT(Length(1, MILE), eq(Length(1760, YARD)));
    }
};
```

2.2.2 面向对象

Magellan 实现 xUnit 时非常巧妙，使得用户设计用例时更加面向对象，例如下例。

示例代码 2-3 test/robot-cleaner/RobotCleanerTest.cpp

```
#include "magellan/magellan.hpp"
#include "robot-cleaner/RobotCleaner.h"
#include "robot-cleaner/Position.h"
#include "robot-cleaner/Instructions.h"

USING_HAMCREST_NS

FIXTURE(RobotCleanerTest)
{
    TEST("at the beginning, the robot should be in at the initial position")
    {
        RobotCleaner robot;

        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, NORTH)));
    }

    TEST("left instruction: 1-times")
    {
        RobotCleaner robot;

        robot.exec(left());

        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, WEST)));
    }

    TEST("left instruction: 2-times")
    {
        RobotCleaner robot;

        robot.exec(left());
        robot.exec(left());

        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, SOUTH)));
    }
};
```

用例之间存在重复代码，为了改善设计，首先将所有用例使用的 RobotCleaner 对象直接定义在类体里即可。但在运行时，每个用例可得到独立的 RobotCleaner 实例。

示例代码 2-4 test/robot-cleaner/RobotCleanerTest.cpp

```
#include "magellan/magellan.hpp"
#include "robot-cleaner/RobotCleaner.h"
#include "robot-cleaner/Position.h"
#include "robot-cleaner/Instructions.h"

USING_HAMCREST_NS

FIXTURE(RobotCleanerTest)
{
    RobotCleaner robot;

    TEST("at the beginning, the robot should be in at the initial position")
    {
        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, NORTH)));
    }

    TEST("left instruction: 1-times")
    {
        robot.exec(left());
        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, WEST)));
    }

    TEST("left instruction: 2-times")
    {
        robot.exec(left());
        robot.exec(left());
        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, SOUTH)));
    }
};
```

2.2.3 提取函数

提取相关的函数，改善用户的表达力。提取的相关子函数，可以放在 Fixture 的内部，使得用例与其的距离最近，更加体现类作用域的概念。

示例代码 2-5 test/robot-cleaner/RobotCleanerTest.cpp

```
#include "magellan/magellan.hpp"
#include "robot-cleaner/RobotCleaner.h"
#include "robot-cleaner/Position.h"
#include "robot-cleaner/Instructions.h"

USING_HAMCREST_NS
```

```
FIXTURE(RobotCleanerTest)
{
    RobotCleaner robot;

    void WHEN_I_send_instruction(Instruction* instruction)
    {
        robot.exec(instruction);
    }

    void AND_I_send_instruction(Instruction* instruction)
    {
        WHEN_I_send_instruction(instruction);
    }

    void THEN_the_robot_cleaner_should_be_in(const Position& position)
    {
        ASSERT_THAT(robot.getPosition(), is(position));
    }

    TEST("at the beginning, the robot should be in at the initial position")
    {
        ASSERT_THAT(robot.getPosition(), is(Position(0, 0, NORTH)));
    }

    TEST("left instruction: 1-times")
    {
        WHEN_I_send_instruction(left());
        THEN_the_robot_cleaner_should_be_in(Position(0, 0, WEST));
    }

    TEST("left instruction: 2-times")
    {
        WHEN_I_send_instruction(repeat(left(), 2));
        THEN_the_robot_cleaner_should_be_in(Position(0, 0, SOUTH));
    }

    TEST("left instruction: 3-times")
    {
        WHEN_I_send_instruction(repeat(left(), 3));
        THEN_the_robot_cleaner_should_be_in(Position(0, 0, EAST));
    }

    TEST("left instruction: 4-times")
    {
        WHEN_I_send_instruction(repeat(left(), 4));
        THEN_the_robot_cleaner_should_be_in(Position(0, 0, NORTH));
    }
};
```

This page is intentionally left blank.

Any fool can write code that a computer can understand. Good programmers write code that humans can understand.

- Martin Flower

3 断言

3.1 ASSERT_THAT

Magellan 只支持一种断言原语：ASSERT_THAT，从而避免用户在 ASSERT_EQ/ASSERT_NE, ASSERT_TRUE/ASSERT_FALSE 之间做选择时的困扰，使其断言更加具有统一性。此外，ASSERT_THAT 使得断言更加具有表达力，更加符合语言习惯。

示例代码 3-1 test/hamcrest/CloseToTest.cpp

```
#include <magellan/magellan.hpp>

FIXTURE(CloseToTest)
{
    TEST("double")
    {
        ASSERT_THAT(1.0, close_to(1.0, 0.5));
        ASSERT_THAT(0.5, close_to(1.0, 0.5));
        ASSERT_THAT(1.5, close_to(1.0, 0.5));
    }
};
```

3.2 Matcher

3.2.1 Anything

匹配器	说明
anything	总是匹配
_	anything 语法糖

表 3.1 anything

示例代码 3-2 test/hamcrest/AnythingTest.cpp

```
#include <magellan/magellan.hpp>

USING_HAMCREST_NS

FIXTURE(AnythingTest)
{
    TEST("should always be matched")
    {
        ASSERT_THAT(1, anything<int>());
        ASSERT_THAT(1u, anything<unsigned int>());
        ASSERT_THAT(1.0, anything<double>());
        ASSERT_THAT(1.0f, anything<float>());
        ASSERT_THAT(false, anything<bool>());
        ASSERT_THAT(true, anything<bool>());
        ASSERT_THAT(nullptr, anything<std::nullptr_t>());
    }

    TEST("should support _ as syntactic sugar")
    {
        ASSERT_THAT(1u, _(int));
        ASSERT_THAT(1.0f, _(float));
        ASSERT_THAT(false, _(int));
        ASSERT_THAT(nullptr, _(std::nullptr_t));
    }
};
```

3.2.2 比较器

匹配器	说明
eq	相等
ne	不相等
lt	小于
gt	大于
le	小于或等于
ge	大于或等于

表 3.2 比较的 Matcher

示例代码 3-3 test/hamcrest/ComparableTest.cpp

```cpp
#include <magellan/magellan.hpp>

USING_HAMCREST_NS

FIXTURE(EqualToTest)
{
    TEST("should allow compare to integer")
    {
        ASSERT_THAT(0xFF, eq(0xFF));
        ASSERT_THAT(0xFF, is(eq(0xFF)));

        ASSERT_THAT(0xFF, is(0xFF));
        ASSERT_THAT(0xFF == 0xFF, is(true));
    }

    TEST("should allow compare to bool")
    {
        ASSERT_THAT(true, eq(true));
        ASSERT_THAT(false, eq(false));
    }

    TEST("should allow compare to string")
    {
        ASSERT_THAT("hello", eq("hello"));
        ASSERT_THAT("hello", eq(std::string("hello")));
        ASSERT_THAT(std::string("hello"), eq(std::string("hello")));
    }
};

FIXTURE(NotEqualToTest)
{
    TEST("should allow compare to integer")
    {
        ASSERT_THAT(0xFF, ne(0xEE));

        ASSERT_THAT(0xFF, is_not(0xEE));
        ASSERT_THAT(0xFF, is_not(eq(0xEE)));
        ASSERT_THAT(0xFF != 0xEE, is(true));
    }

    TEST("should allow compare to boolean")
    {
        ASSERT_THAT(true, ne(false));
        ASSERT_THAT(false, ne(true));
    }

    TEST("should allow compare to string")
    {
        ASSERT_THAT("hello", ne("world"));
        ASSERT_THAT("hello", ne(std::string("world")));
        ASSERT_THAT(std::string("hello"), ne(std::string("world")));
    }
};
```

3.2.3 修饰器

匹配器	说明
is	可读性装饰器
is_not	可读性装饰器

表 3.3 修饰的 Matcher

示例代码 3-4 test/hamcrest/IsNotTest.cpp

```
#include <magellan/magellan.hpp>

USING_HAMCREST_NS

FIXTURE(IsNotTest)
{
    TEST("integer")
    {
        ASSERT_THAT(0xFF, is_not(0xEE));
        ASSERT_THAT(0xFF, is_not(eq(0xEE)));
    }

    TEST("string")
    {
        ASSERT_THAT("hello", is_not("world"));
        ASSERT_THAT("hello", is_not(eq("world")));

        ASSERT_THAT("hello", is_not(std::string("world")));
        ASSERT_THAT(std::string("hello"), is_not(std::string("world")));
    }
};
```

3.2.4 空指针

匹配器	说明
nil	空指针

表 3.4 空指针

示例代码 3-5 test/hamcrest/NilTest.cpp

```
#include <magellan/magellan.hpp>

USING_HAMCREST_NS

FIXTURE(NilTest)
{
    TEST("equal_to")
    {
        ASSERT_THAT(nullptr, eq(nullptr));
        ASSERT_THAT(0, eq(NULL));
        ASSERT_THAT(NULL, eq(NULL));
        ASSERT_THAT(NULL, eq(0));
    }

    TEST("is")
    {
        ASSERT_THAT(nullptr, is(nullptr));
        ASSERT_THAT(nullptr, is(eq(nullptr)));

        ASSERT_THAT(0, is(0));
        ASSERT_THAT(NULL, is(NULL));
        ASSERT_THAT(0, is(NULL));
        ASSERT_THAT(NULL, is(0));
    }

    TEST("nil")
    {
        ASSERT_THAT((void*)NULL, nil());
        ASSERT_THAT((void*)0, nil());
        ASSERT_THAT(nullptr, nil());
    }
};
```

3.2.5 字符串

示例代码 3-6 test/hamcrest/StartsWithTest.cpp

```
#include <magellan/magellan.hpp>

USING_HAMCREST_NS

FIXTURE(StartsWithTest)
{
    TEST("case sensitive")
    {
        ASSERT_THAT("ruby-cpp", starts_with("ruby"));
        ASSERT_THAT("ruby-cpp", is(starts_with("ruby")));

        ASSERT_THAT(std::string("ruby-cpp"), starts_with("ruby"));
        ASSERT_THAT("ruby-cpp", starts_with(std::string("ruby")));
```

匹配器	说明
contains_string	断言是否包含子串
contains_string_ignoring_case	忽略大小写，断言是否包含子串
starts_with	断言是否以该子串开头
starts_with_ignoring_case	忽略大小写，断言是否以该子串开头
ends_with	断言是否以该子串结尾
ends_with_ignoring_case	忽略大小写，断言是否以该子串结尾

表 3.5 字符串的 Matcher

```
        ASSERT_THAT(std::string("ruby-cpp"), starts_with(std::string("ruby")));
    }

    TEST("ignoring case")
    {
        ASSERT_THAT("ruby-cpp", starts_with_ignoring_case("Ruby"));
        ASSERT_THAT("ruby-cpp", is(starts_with_ignoring_case("Ruby")));

        ASSERT_THAT(std::string("ruby-cpp"), starts_with_ignoring_case("RUBY"));
        ASSERT_THAT("Ruby-Cpp", starts_with_ignoring_case(std::string("rUBY")));
        ASSERT_THAT(std::string("RUBY-CPP"), starts_with_ignoring_case(std::string("ruby")));
    }
};
```

3.2.6 浮点数

匹配器	说明
close_to	断言浮点数近似等于
nan	断言浮点数不是一个数字

表 3.6 浮点数的 Matcher

示例代码 3-7 test/hamcrest/NanTest.cpp

```
#include <magellan/magellan.hpp>
#include <math.h>

USING_HAMCREST_NS

FIXTURE(IsNanTest)
{
    TEST("double")
    {
        ASSERT_THAT(sqrt(-1.0), nan());
        ASSERT_THAT(sqrt(-1.0), is(nan()));

        ASSERT_THAT(1.0/0.0,  is_not(nan()));
        ASSERT_THAT(-1.0/0.0, is_not(nan()));
    }
};
```