# CCTEST: 现代 C++ 测试框架的演进之路

## A Simple xUnit Implement in Modern C++

刘 光聪

horance@aliyun.com

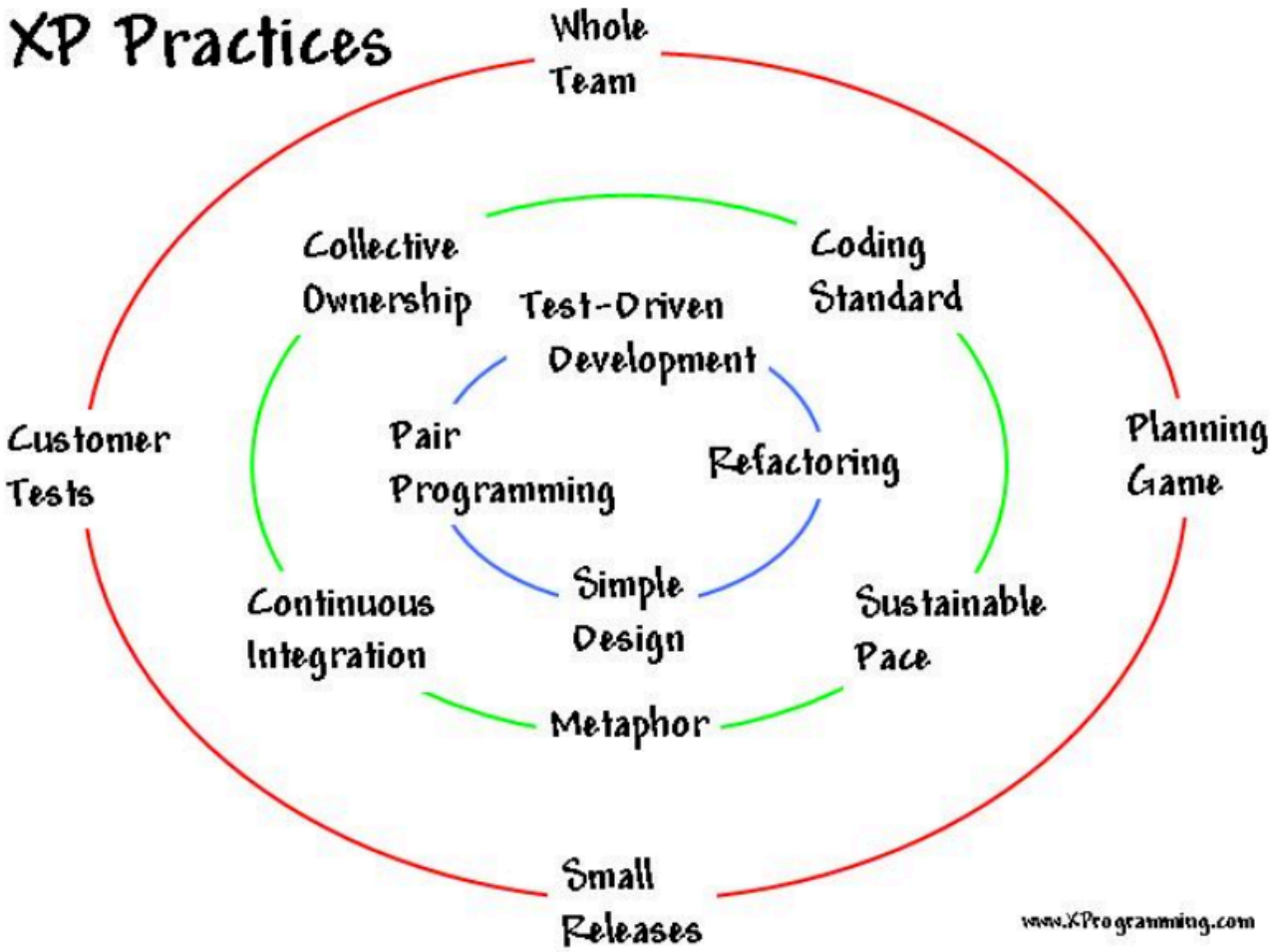
2019.10

# 内容

- 动机
- 特性介绍
- 设计实现
- 未来演进

# 动机

# 极限编程


XP Practices
Whole Team
Collective Ownership
Coding Standard
Test-Driven Development
Pair Programming
Refactoring
Customer Tests
Planning Game
Continuous Integration
Simple Design
Sustainable Pace
Metaphor
Small Releases
www.XProgramming.com

# 测试 vs. TDD

**传统UT: 一种事后的测试活动。**

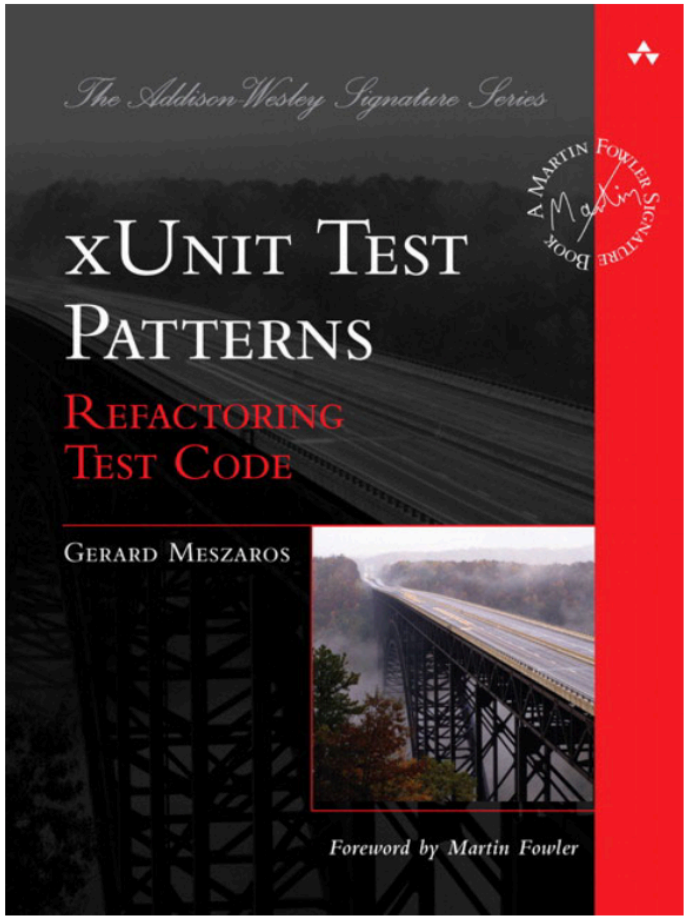
- 事后测试;
- 实现与测试高度耦合: 用例复杂且脆弱;
- 以验证为目标, 无法保障设计的改进, 甚至阻碍设计的改进。

**TDD(Test Driven Development/Design): 一种开发/设计活动, 作为简单设计的衡量标尺。**

- 改进设计: 测试是手段, 设计是目标;
- 设计文档+使用手册: 塑形设计;
- 反馈系统: 快速反馈, 构建重构的安全网。

# xUnit 实现模式

- Tests as Specification
- Tests as Document
- Tests as Example
- Tests as Safety Net
- Tests as Design Tool



# 回到主题：为什么需要一个全新的 C/C++ 测试框架？

让 TDD 和重构更高效，让编程添加更多乐趣

# 我的偏爱

1. C/C++: GoogleTest/Catch2
2. Java: JUnit/Spock
3. Python: unittest
4. Ruby: RSpec
5. Scala: ScalaTest

# 命名困难

**坏味道：使用标识符命名用例**

```
// Bad Smell: test cases must be named using c++ identifier.
TEST_F(RobotCleanerTest, at_beginning_the_robot_should_be_in_at_the_initial_position) {
  ASSERT_EQ(Position(0, 0, NORTH), robot.getPosition());
}
```

# 重复设计

## 重复设计：测试装置与测试用例相分离

```
// Bad Smell: you must duplicate fixture name for each test case.
struct RobotCleanerTest : testing::Test {
protected:
  RobotCleaner robot;
};

TEST_F(RobotCleanerTest, at_beginning_the_robot_should_be_in_at_the_initial_position) {
  ASSERT_EQ(Position(0, 0, NORTH), robot.getPosition());
}

TEST_F(RobotCleanerTest, robot_should_be_face_west_after_turn_left) {
  robot.turnLeft();
  ASSERT_EQ(Position(0, 0, WEST), robot.getPosition());
}
```

# 隐式继承

**错误：本应该使用 protected，而错误地使用 private**

```
struct RobotCleanerTest : testing::Test {
private: // should be protected
  RobotCleaner robot;
};

TEST_F(RobotCleanerTest, at_beginning_the_robot_should_be_in_at_the_initial_position) {
  // Error: 'RobotCleaner RobotCleanerTest::robot' is private within this context.
  ASSERT_EQ(Position(0, 0, NORTH), robot.getPosition());
}
```

# 易于误用 1

**错误：本应该使用 TEST_F，而错误地使用 TEST**

```
struct RobotCleanerTest : testing::Test {
protected:
  RobotCleaner robot;
};

// should be TEST_F
TEST(RobotCleanerTest, at_beginning_the_robot_should_be_in_at_the_initial_position) {
  // Error: 'robot' was not declared in this scope.
  ASSERT_EQ(Position(0, 0, NORTH), robot.getPosition());
}
```

# 易于误用 2

**错误：本应该声明为 SetUp，而错误地声明为 Setup**

```
struct RobotCleanerTest : testing::Test {
private:
  // Error: should override SetUp, not Setup/setup/setUp.
  void Setup() {
    robot.reset();
  }

protected:
  RobotCleaner robot;
};
```

# 特性介绍

# 需求：单位换算

1. 1 FEET == 12 INCH
2. 1 YARD == 3 FEET
3. 1 MILE == 1760 YARD

# 测试用例

```
#include "quantity/length.h"
#include "cctest/cctest.h"

namespace {

FIXTURE(LengthTest) {
  TEST("1 feet == 12 inch") {
    ASSERT_EQ(Length(1, FEET), Length(12, INCH));
  }

  TEST("1 yard == 3 feets") {
    ASSERT_EQ(Length(1, YARD), Length(3, FEET));
  }

  TEST("1 mile == 1760 yards") {
    ASSERT_EQ(Length(1, MILE), Length(1760, YARD));
  }
};

} // namespace
```

# quantity/length.h

```
#include "quantity/amount.h"
#include "quantity/length_unit.h"

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

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

private:
  Amount amountInBaseUnit;
};
```

# quantity/length_unit.h

```
enum LengthUnit {
  INCH = 1,
  FEET = 12 * INCH
};
```

# quantity/amount.h

```
using Amount = unsigned int;
```

# quantity/length.cc

```
#include "quantity/length.h"

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

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

# 自我测试

```
" Press ? for help
.. (up a dir)
</wisdomcoda/cctest/
▸bazel-bin/ -> /private/
▸bazel-cctest/ -> /priva
▸bazel-genfiles/ -> /pri
▸bazel-out/ -> /private/
▸bazel-testlogs/ -> /pri
▸build/
▸cctest/
▸cmake/
▸dockerfiles/
▸examples/
▾spec/
  ▸auto/
  ▸base/
  ▸core/
  ▾listener/
      BUILD.bazel
      CMakeLists.txt
      failure_list_spec.c
      test_collector_spec
      test_status_spec.cc
      text_progress_spec.
      time_collector_spec
  ▾matcher/
      CMakeLists.txt
      description_spec.cc
    CMakeLists.txt
  CMakeLists.txt
  commit.log
  install.sh
  LICENSE
  README.md
  WORKSPACE
```

```
 1 #include "cctest/cctest.h"
 2 #include "cctest/listener/collector/test_status.h"
 3 #include "cctest/core/test_result.h"
 4
 5 using namespace cctest;
 6
 7 namespace {
 8
 9 FIXTURE(TestStatusSpec) {
10   TestResult result;
11   TestStatus status;
12
13   SETUP {
14     result.addListener(status);
15   }
16
17   void run(cctest::Test& test) {
18     test.run(result);
19   }
20
21   struct FailureOnRunningTest : TestCase {
22   private:
23     void runTest() override {
24       throw AssertionError("product.cc:57", "expected value == 2, but got 3");
25     }
26   };
27
28   TEST("should be failed") {
29     FailureOnRunningTest test;
30     run(test);
31     ASSERT_FALSE(status.successful());
32   }
33 };
34
35 } // namespace
~
```

# 报表定制化

```
horance:~/code/wisdomcoda/cctest/build(master ✗) »spec/listener/cctest_test_collector_spec
[==========] 4 test cases.
[----------] 4 tests from TestCollectorSpec
[ RUN      ] run one simple test
[       OK ] run one simple test(15 us)
[ RUN      ] throw assertion error on run test
[       OK ] throw assertion error on run test(67 us)
[ RUN      ] throw std exception on run test
[       OK ] throw std exception on run test(21 us)
[ RUN      ] count test cases from collector
[       OK ] count test cases from collector(23 us)
[----------] 4 tests from TestCollectorSpec(190 us)
[==========] 4 test cases.
[  TOTAL   ] PASS: 4  FAILURE: 0  ERROR: 0  TIME: (554 us)
```

- 标准输出：彩色终端，纯色终端
- CI/CD 报表：XML, JSON, YAML

# 构建系统: Bazel

**WORKSPACE: 声明外部依赖**

```
# using cctest for xunit test framework.
# github: https://github.com/ccup/cctest
# curl -L https://github.com/ccup/cctest/archive/1ce2bed5cbf7e9a282eee546ff5637a040e42abe.tar.gz | sha256sum
http_archive(
    name = "xunit_cctest",
    sha256 = "f7c2c339a5ab06dc1d16cb03b157a96e6c591f9833f5c072f56af4a8f8013b53",
    strip_prefix = "cctest-1ce2bed5cbf7e9a282eee546ff5637a040e42abe",
    urls = [
        "https://github.com/ccup/cctest/archive/1ce2bed5cbf7e9a282eee546ff5637a040e42abe.tar.gz",
    ],
)
```

**BUILD.bazel: 定义测试目标**

```
cc_library(
    name = "check_sum",
    hdrs = ["check_sum.h"],
    srcs = ["check_sum.cc"],
    deps = [":str_utils"],
)

cc_test(
    name = "check_sum_test",
    srcs = ["check_sum_test.cc"],
    deps = [
        ":check_sum",
        "@xunit_cctest//cctest",
        "@xunit_cctest//cctest:main",
    ],
)
```

**$ bazel test //spec/...: 执行所有用例**

```
horance:~/code/wisdomcoda/cctest/build(master ⚡) »bazel test //spec/...
Starting local Bazel server and connecting to it...
INFO: Analyzed 14 targets (23 packages loaded, 368 targets configured).
INFO: Found 14 test targets...
INFO: Deleting stale sandbox base /private/var/tmp/_bazel_horance/b158852d0094774cffa97e6b5254c0df/sandbox
INFO: Elapsed time: 19.917s, Critical Path: 3.88s
INFO: 84 processes: 84 darwin-sandbox.
INFO: Build completed successfully, 101 total actions
//spec/auto:queue_spec                                  PASSED in 0.1s
//spec/auto:stack_spec                                  PASSED in 0.1s
//spec/core:asserter_spec                               PASSED in 0.1s
//spec/core:test_case_spec                              PASSED in 0.1s
//spec/core:test_desc_spec                              PASSED in 0.1s
//spec/core:test_fixture_spec                           PASSED in 0.1s
//spec/core:test_method_spec                            PASSED in 0.1s
//spec/core:test_result_spec                            PASSED in 0.2s
//spec/core:test_suite_spec                             PASSED in 0.1s
//spec/listener:failure_list_spec                       PASSED in 0.1s
//spec/listener:test_collector_spec                     PASSED in 0.1s
//spec/listener:test_status_spec                        PASSED in 0.1s
//spec/listener:text_progress_spec                      PASSED in 0.1s
//spec/listener:time_collector_spec                     PASSED in 0.1s

Executed 14 out of 14 tests: 14 tests pass.
INFO: Build completed successfully, 101 total actions
```

# 构建系统: CMake

**CMakeLists.txt: 定义测试目标**

```
cc_test(
  NAME
    asserter_spec
  SRCS
    asserter_spec.cc
)

cc_test(
  NAME
    test_method_spec
  SRCS
    test_method_spec.cc
)

cc_test(
  NAME
    test_case_spec
  SRCS
    test_case_spec.cc
)

cc_test(
  NAME
    test_fixture_spec
  SRCS
    test_fixture_spec.cc
)

cc_test(
```

**$ cmake .. && make: 构建被测系统**

```
horance:~/code/wisdomcoda/cctest(master 1)» mkdir build && cd build && cmake ..
-- The CXX compiler identification is AppleClang 10.0.0.10001044
-- Check for working CXX compiler: /Library/Developer/CommandLineTools/usr/bin/c++
-- Check for working CXX compiler: /Library/Developer/CommandLineTools/usr/bin/c++
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- No CMAKE_CXX_STANDARD set, assuming 11
-- Configuring done
-- Generating done
-- Build files have been written to: /Users/horance/code/wisdomcoda/cctest/build
horance:~/code/wisdomcoda/cctest/build(master 2)» make
Scanning dependencies of target cctest
[  1%] Building CXX object cctest/CMakeFiles/cctest.dir/base/color.cc.o
[  3%] Building CXX object cctest/CMakeFiles/cctest.dir/base/formatter.cc.o
[  4%] Building CXX object cctest/CMakeFiles/cctest.dir/base/logger.cc.o
[  6%] Building CXX object cctest/CMakeFiles/cctest.dir/base/string_utils.cc.o
[  8%] Building CXX object cctest/CMakeFiles/cctest.dir/base/string_view.cc.o
```

**$ make test: 执行所有用例**

```
horance:~/code/wisdomcoda/cctest/build(master 3)» make test
Running tests...
Test project /Users/horance/code/wisdomcoda/cctest/build
      Start  1: cctest_string_view_spec
 1/17 Test  #1: cctest_string_view_spec .....   Passed    0.01 sec
      Start  2: cctest_string_utils_spec
 2/17 Test  #2: cctest_string_utils_spec .....   Passed    0.01 sec
      Start  3: cctest_stack_spec
 3/17 Test  #3: cctest_stack_spec ............   Passed    0.02 sec
      Start  4: cctest_queue_spec
 4/17 Test  #4: cctest_queue_spec ............   Passed    0.02 sec
```

# Anotation 语义

## 语义定制

1. 测试依赖
2. 超时检测
3. 异常检测
4. 执行数目
5. Tag 标签过滤
6. 内存溢出检测

**Transaction DSL案例**

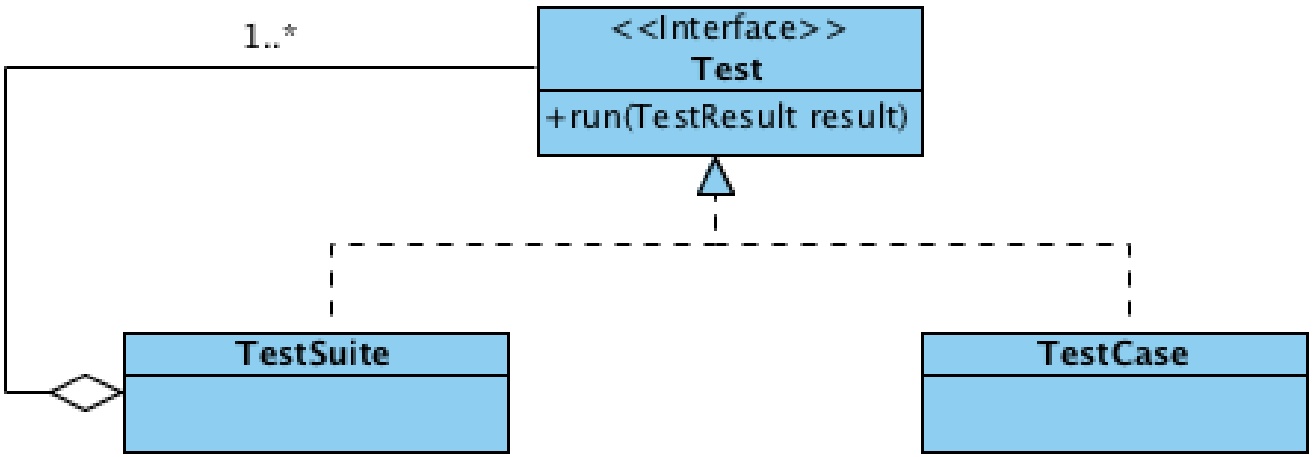
```
40 FIXTURE("@{tags=seq} sequential") {   Tag Annotation
41     __transaction
42     ( __sequential
43         ( __wait(1)
44         , __wait(2))
45     )trans;
46
47     SETUP {
48         ASSERT_EQ(TSL_CONTINUE, trans.start());
49     }
50           Test Identity
51     TEST("@{id=event-1} after recv event-1, should return TSL_CONTINUE") {
52         ASSERT_EQ(TSL_CONTINUE, trans.handleEvent(EVENT(1)));
53     }
54
55     TEST("after recv event-2, should return UNKNOWN_EVENT") {
56         ASSERT_EQ(TSL_UNKNOWN_EVENT, trans.handleEvent(EVENT(2)));
57     }
58         Test Dependency
59     TEST("@{dep=event-1} after recv event-1, if recv event-2, should return TSL_SUCCESS") {
60         ASSERT_EQ(TSL_SUCCESS, trans.handleEvent(EVENT(2)));
61     }
62
63     TEST("after kill, if recv event-1, should return UNKNOWN_EVENT")
64     {
65         trans.kill();
66         ASSERT_EQ(TSL_UNKNOWN_EVENT, trans.handleEvent(EVENT(1)));
67     }
68 };
```

# 可扩展断言机制：ASSERT_THAT & 匹配器

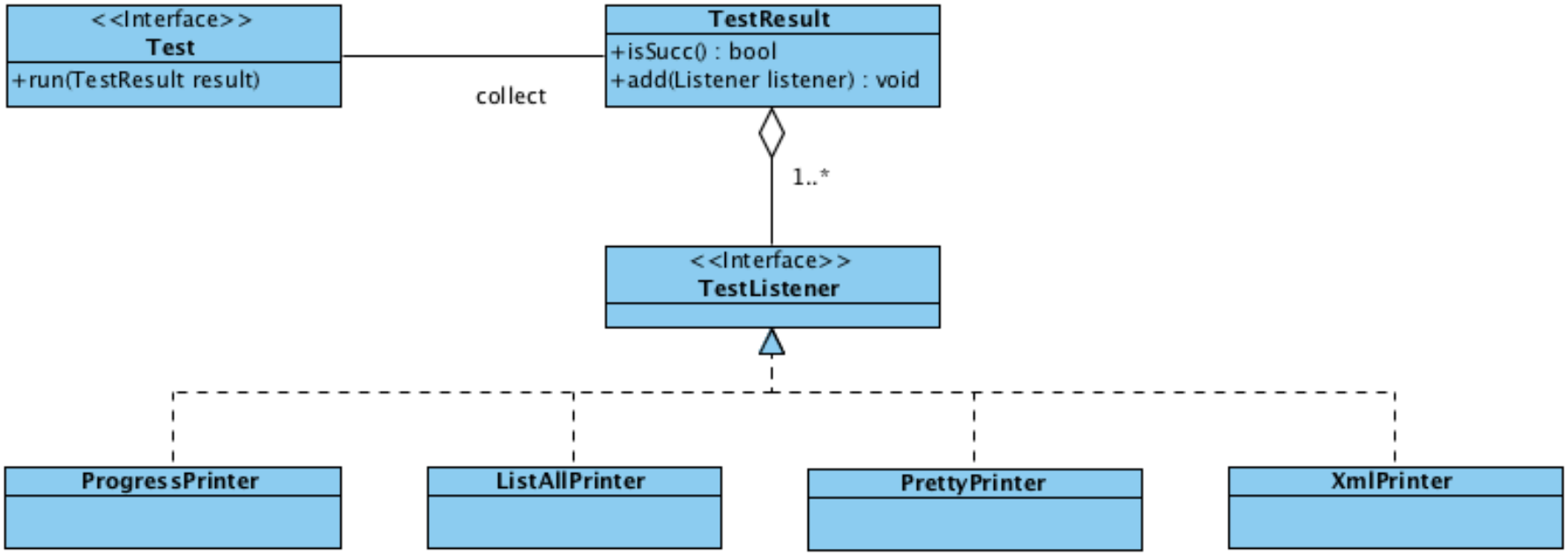
```
 7 FIXTURE(StartsWithTest) {
 8     TEST("case sensitive") {
 9         ASSERT_THAT("ruby-cpp", starts_with("ruby"));
10         ASSERT_THAT("ruby-cpp", is(starts_with("ruby")));
11 
12         ASSERT_THAT(std::string("ruby-cpp"), starts_with("ruby"));
13         ASSERT_THAT("ruby-cpp", starts_with(std::string("ruby")));
14         ASSERT_THAT(std::string("ruby-cpp"), starts_with(std::string("ruby")));
15     }
16 
17     TEST("ignoring case") {
18         ASSERT_THAT("ruby-cpp", starts_with_ignoring_case("Ruby"));
19         ASSERT_THAT("ruby-cpp", is(starts_with_ignoring_case("Ruby")));
20 
21         ASSERT_THAT(std::string("ruby-cpp"), starts_with_ignoring_case("RUBY"));
22         ASSERT_THAT("Ruby-Cpp", starts_with_ignoring_case(std::string("rUBY")));
23         ASSERT_THAT(std::string("RUBY-CPP"), starts_with_ignoring_case(std::string("ruby")));
24     }
25 };
```

# 设计实现

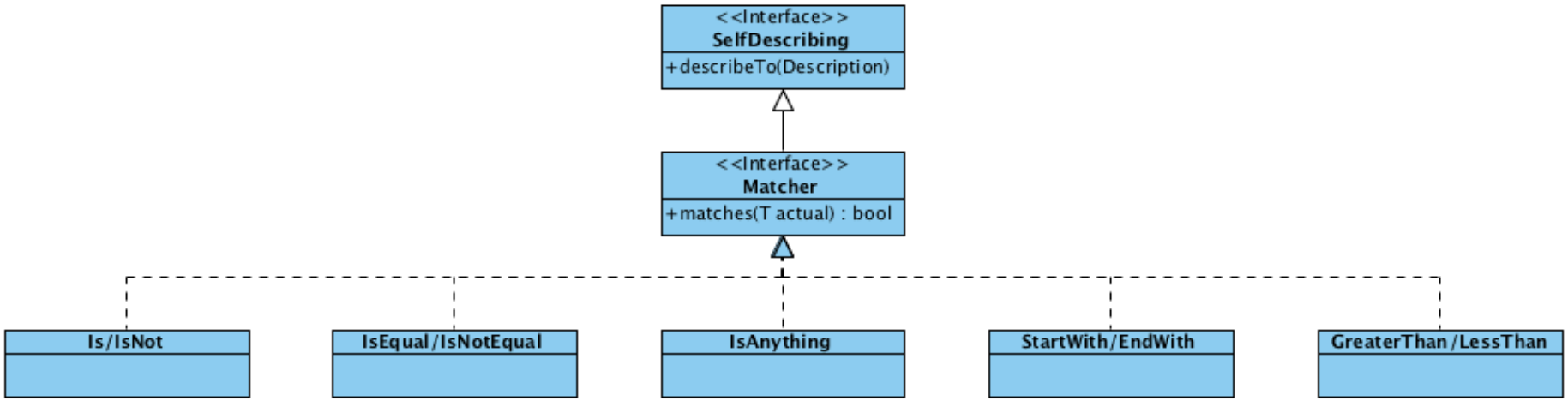
# 核心领域模型


1..*
<<Interface>>
Test
+run(TestResult result)
TestSuite
TestCase

# 监听器


<<Interface>>
Test
+run(TestResult result)
collect
TestResult
+isSucc() : bool
+add(Listener listener) : void
1..*
<<Interface>>
TestListener
ProgressPrinter
ListAllPrinter
PrettyPrinter
XmlPrinter

# 匹配器


<<Interface>>
SelfDescribing
+describeTo(Description)
<<Interface>>
Matcher
+matches(T actual) : bool
Is/IsNot
IsEqual/IsNotEqual
IsAnything
StartWith/EndWith
GreaterThan/LessThan

# 未来演进

# 演进

- BDD 风格
- 沙盒运行时
- 参数化测试
- ...

# 源代码

- GitHub: https://github.com/ccup/cctest
- Slides: https://github.com/horance-liu/cctest-slides

# 谢谢