Статический анализ кода

На прошлой лекции

Основные инструменты построения абстракций:

- Объединение данных и методов (классы)
- Обобщение (шаблоны)

Обзор стандартной библиотеки:

- Контейнеры
- Алгоритмы

Важные детали: константность, вывод типов.

Правила, правила…

Используйте range-based for loop, если не нужен индекс.

```
for (std::size_t i = 0; i < v.size(); ++i)
  // use v[i]

for (const auto& item : v)
  // use item
```

Правила, правила…

NB: operator[] модифицирует контейнер.

```
std::map<std::string, int> text_to_number;
// Если ключа нет, добавит элемент {“two”, 0}.
auto v1 = text_to_number[“two”];

// [...] бросит исключение std::out_of_range
auto v2 = text_to_number.at(“one”);

// [...] вернет text_to_number.end()
auto it = text_to_number.find(“three”);
```

Guidelines

Уже при беглом знакомстве мы встречаем множество рекомендаций.

Q1: Где искать лучшие практики?

Q2: Как следить за их соблюдением?

Guidelines

-Wall -Wextra -Werror -pedantic

C++ Core Guidelines

Clang Tidy

Scott Meyers Book Series, -Weffc++

Предупреждения компилятора

`-Wall -Wextra` — подходит для большинства проектов
`-pedantic` — проверка на строгое соответствие стандарту

Могут быть полезны отдельные проверки:
`-Wold-style-cast, -Wdocumentation (Clang)`

Посмотреть включенные диагностики:
`g++ -Q --help=warnings -Wall`

Пример -pedantic

```
int main() {
  int x[0];
}

In function 'int main()':
ISO C++ forbids zero-size array 'x' [-Wpedantic]
    2 |     int x[0];
      |
```

Non-Standard: https://godbolt.org/z/jq9Knrn8q
Standard: https://godbolt.org/z/rfjGPsr5r

Пример -Wold-style-cast

```
auto* x = (int*)malloc(sizeof(int));
warning: use of old-style cast to 'int*'
```

```
auto* x = static_cast<int*>(malloc(sizeof(int)));
```

C++-Style Cast Operators

```
static_cast
dynamic_cast
reinterpret_cast
const_cast
```

Мотивация:

1. Разный смысл
2. Легко находить в коде

C-Style Cast

Запись вида (T)x выполняет приведение в следующем порядке:

1. const_cast
2. static_cast
3. static_cast + const_cast,
4. reinterpret_cast
5. reinterpret_cast + const_cast

[expr.cast]

Пример -Wdocumentation (Clang)

```
//! \param y value
double square(double x);

warning: parameter 'y' not found [...]
//! \param y value
            ^
note: did you mean 'x'?
//! \param y value
            ^
            x
```

Список диагностик

```
g++ -Q --help=warnings -Wall

...
-Wabsolute-value                    [disabled]
-Waddress                           [enabled]
-Waddress-of-packed-member          [enabled]
-Waggregate-return                  [disabled]
...
```

C++ Core Guidelines

In.target: Target readership

All C++ programmers.

https://isocpp.github.io/CppCoreGuidelines/CppCoreGuidelines

Clang Tidy

clang-tidy — статический анализатор кода.

- 400+ диагностик
  - `clang-tidy -checks='*' -list-checks | wc -l`
- Интеграция с CMake

Clang Tidy: группы проверок

Project-specific:

- abseil-*
- android-*
- …

Common:

- readability-*
- modernize-*
- bugprone-*
- **cppcoreguidelines**-*

Не все проверки одинаково полезны

Clang-Tidy Checks

NB: не теряйте диагностики!

```
#include <libmath/sum.hpp>
#include <cxxopts/cxxopts.h>
#include <vector>
```

Файлы вашего проекта

Сторонние

Стандартные

.clang-tidy:

```
HeaderFilterRegex: '(libmath|app)'
```

Внедрение в новый проект

Быстрый способ:

- Включить все проверки
- Отключать неподходящие по мере появления

Честный:

- Прочитать всю документацию
- Принять решение по каждой проверке

Внедрение в легаси проект

Радикально:

- Включить все проверки
- Прогнать их на кодовой базе
- Отключить неподходящие
- Исправить в коде полезные
- Загрузить одним коммитом

✗ Пожелать счастливого мержа всем остальным

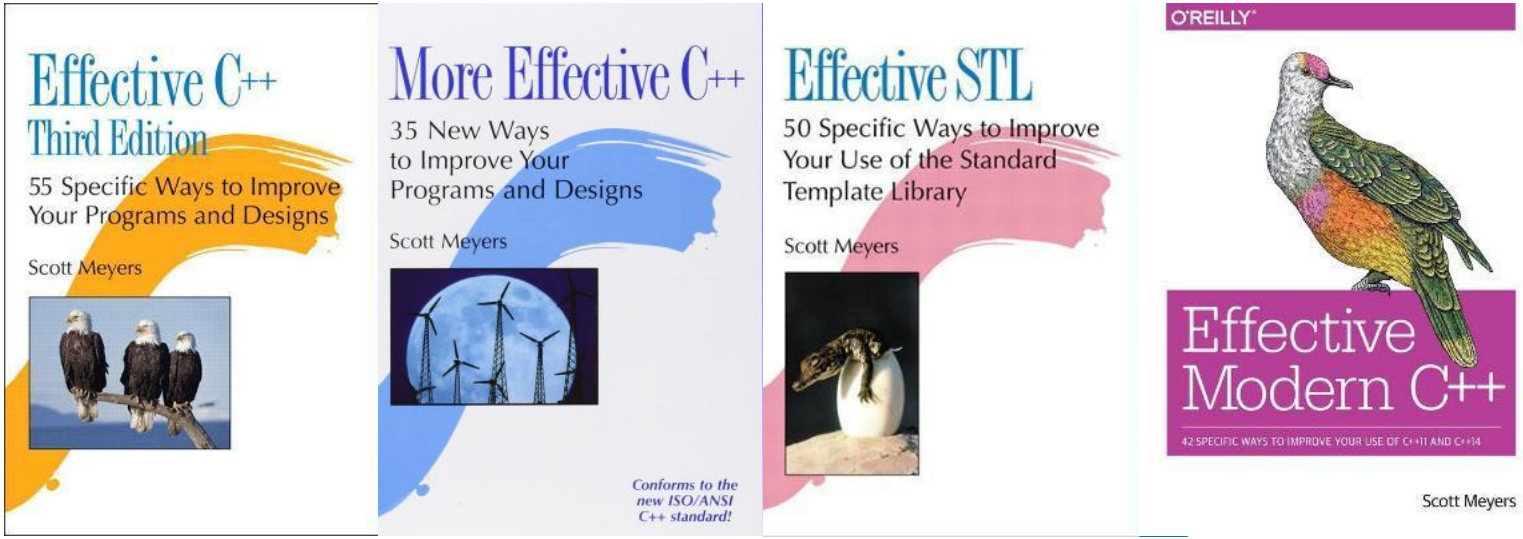
Внедрение в легаси проект

Постепенно:

- Включить все проверки
- Прогнать их на кодовой базе
- Отключить **в конфиге** неподходящие
- Отключить **в коде** полезные
- Исправлять по мере модификации кода

```
// NOLINTNEXTLINE(readability-identifier-length)
double sum(double a, double b);
```

Scott Meyers Book Series



В декабре 2015 Скотт Майерс оставил C++

https://scottmeyers.blogspot.com/2015/12/good-to-go.html

What should we do?

a) That's life: Leave it as it is

b) Require "before/after tables" and style guides for all proposals of C++ features

c) Establish an official style guide as part of the standardization

d) Un-retire Scott Meyers to write a new book or establish another new style guru

e) Improve the standard step-by-step
   - e.g., replace `class` by `typename`

f) Concentrate on safety critical contexts

When C++ Style Guides Contradict

gcc: `-Weffc++`

man gcc:

Warn about violations of the following style guidelines from Scott Meyers' Effective C++ series of books

When selecting this option, be aware that the standard library headers do not obey all of these guidelines; **use grep -v to filter out those warnings**.

Заключение

Статический анализ позволяет:

- Обнаруживать ошибки до запуска приложения
- Контролировать соблюдение соглашений

Не является серебряной пулей