Тестирование ПО

Статическое тестирование

Кулаков Кирилл Александрович

Статическое тестирование

- Статическое тестирование — анализ формальными методами без выполнения тестируемой программы неверных конструкций или неверных отношений объектов программы (ошибки формального задания)
  - ручное исполнение
  - компиляторы
  - среда программирования
  - специальные инструменты

Цели и задачи

- Цель: выявление расхождений на раннем этапе в различных представлениях проекта (код, документация, …)
- Исходные данные:
  - ТЗ
  - Стандарты, спецификации, гайдлайны
  - Проектные ожидания
- Задачи:
  - анализ представления
  - поиск несоответствий между текущим состоянием и ожиданием (например, между кодом и стандартом кодирования)
  - локализация проблемного участка

Ручное тестирование

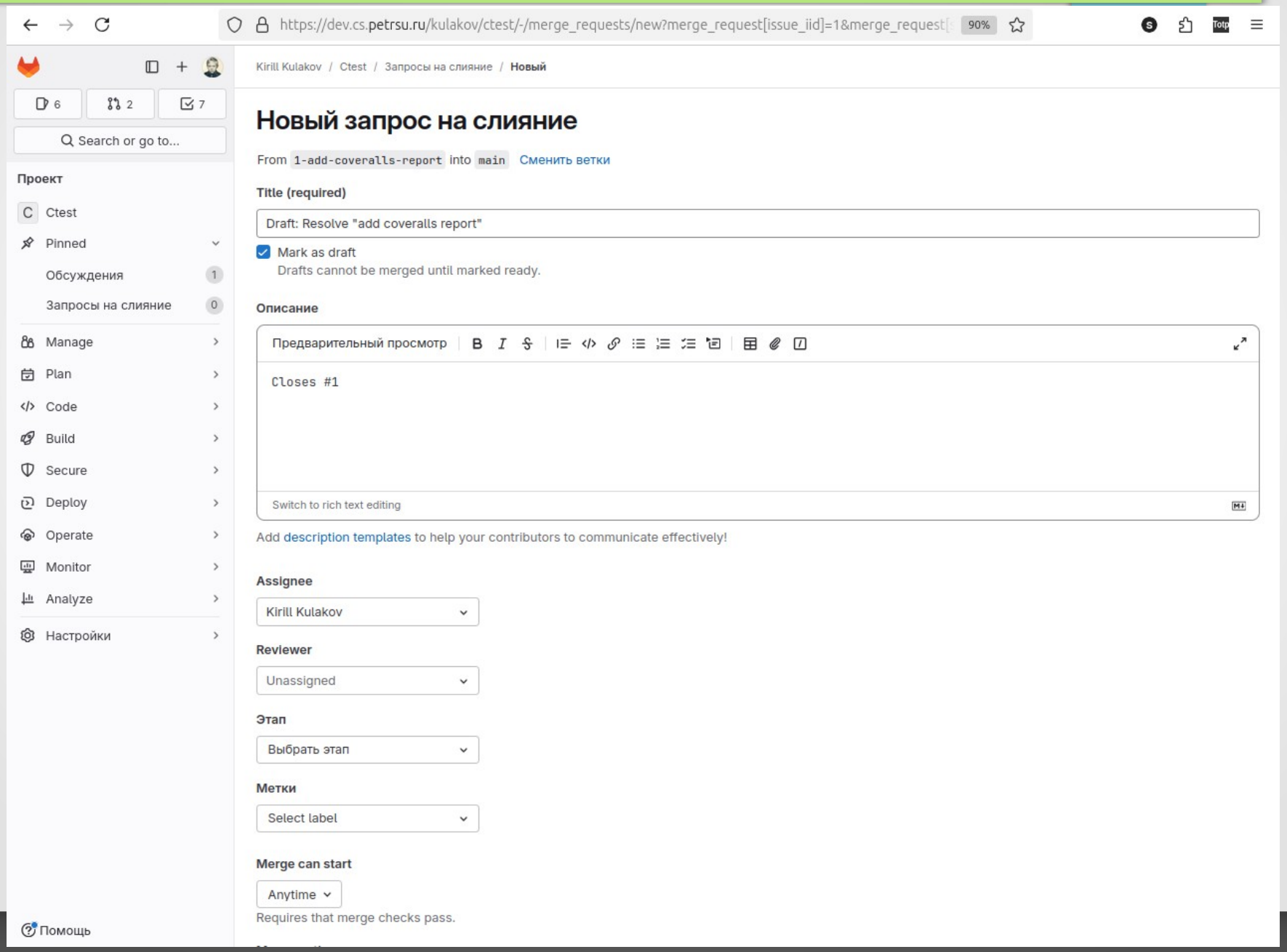
- Выбирается объект тестирования (например, код модуля)
- Назначается ответственный за тестирование
- Проводится ручной анализ объекта (просмотр кода)
- Результаты оформляются в протокол

- Плюсы:
  - минимум ложных срабатываний
  - учет множества критериев
- Минусы:
  - очень долго и дорого
  - человеческий фактор

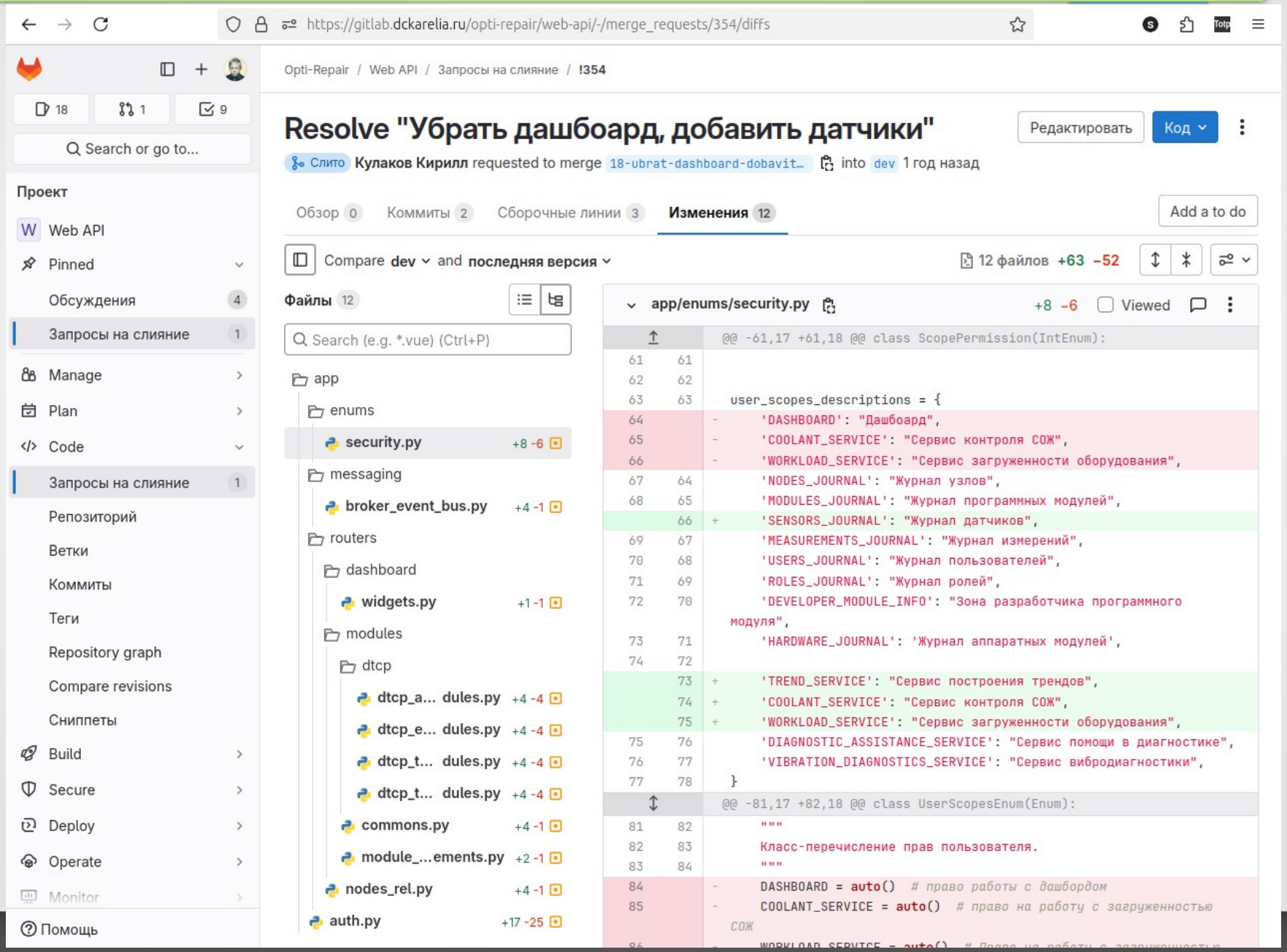
Ручное тестирование

- Критерии оценки: набор(ы) правил
- Пример: правила Скотта Майерса (*Скотт Майерс - Эффективное использование C++. 55 верных способов улучшить структуру и код ваших программ*)
  - Правило 2: Предпочитайте const, enum и inline использованию #define
  - Правило 3: Везде, где только можно используйте const
  - Правило 4: Прежде чем использовать объекты, убедитесь, что они инициализированы
  - Правило 7: Объявляйте деструкторы виртуальными в полиморфном базовом классе
  - Правило 9: Никогда не вызывайте виртуальные функции в конструкторе или деструкторе
  - Правило 18: Проектируйте интерфейсы так, что их легко было использовать правильно и трудно – неправильно
  - ...

Ручное тестирование



Ручное тестирование



Компилятор

- Ошибки компиляции
- Предупреждения
- Опции компиляции
  - -Werr='тип предупреждения' — воспринимать предупреждение как ошибку. Для настроящих самураев -Werr без параметров или -Werror
  - -Wpedantic строгое соответствие стандартам ISO C и ISO C++
  - -Wall включение предупреждений
  - -Wextra включение дополнительных предупреждений

Компилятор (предупреждения)

- **-Waddress** — недопустимое использование адресов памяти
- **-Warray-bounds** — выход за пределы границ массива
- **-Wbool-compare** — сравнение логического и целого
- **-Wbool-operation** — сомнительные операции с логическим типом (например, битовое отрицание)
- **-Wc++11-compat -Wc++14-compat** — использование конструкций, значение которых поменялось (например, ключевые слова в ISO C ++ 2011)
- **-Wcatch-value** — несоответствие обработчика
- **-Wchar-subscripts** — индекс массива типа char
- **-Wcomment** — наличие комментария в комментарии
- **-Wduplicate-decl-specifier** — дублирование спецификатора в декларации
- **-Wenum-compare** — сравнение значений различных типов

Компилятор (предупреждения)

- **-Wformat** — неправильный формат в printf/scanf
- **-Wint-in-bool-context** — использование целочисленного значения вместо логического
- **-Wimplicit** — аналог -Wimplicit-int и -Wimplicit-function-declaration
- **-Wimplicit-int** — декларация не указывает тип
- **-Wimplicit-function-declaration** — использование функции перед объявлением
- **-Winit-self** — самоинициализация (int i=i;)
- **-Wlogical-not-parentheses** — левый операнд не является логическим типом
- **-Wmain** — функция main определена некорректно
- **-Wmaybe-uninitialized** — есть возможность пропустить инициализацию переменной
- **-Wmemset-elt-size** — подозрительный вызов memset (неправильные размеры)

Компилятор (предупреждения)

- **-Wmemset-transposed-args** — подозрительный вызов memset (неправильный порядок аргументов)
- **-Wmisleading-indentation** — одинаковый отступ для строк разных блоков
- **-Wmissing-attributes** — отсутствие атрибутов в объявлении функции
- **-Wmissing-braces** — инициализатор агрегата или объединения не полностью заключен в скобки
- **-Wmultistatement-macros** — небезопасные макросы макросов
- **-Wnonnull** — передача нулевого указателя для аргументов с атрибутом nonnull
- **-Wnonnull-compare** — сравнение nonnull атрибута с null
- **-Wopenmp-simd** — модель стоимости вектора переопределяет директиву OpenMP simd
- **-Wparentheses** — возможно опущены круглые скобки там где они необходимы (например, для указания порядка операций)

Компилятор (предупреждения)

- **-Wpointer-sign** — передача или присвоение с другим типом
- **-Wreorder** — измененный порядок инициализации элементов
- **-Wrestrict** — аргумент может перекрыться другим аргументом (например, копирование части строки внутрь себя)
- **-Wreturn-type** — возвращаемое значение не соответствует ожидаемому
- **-Wsequence-point** — предупреждение о коде с неопределенной семантикой (например, a=a++;)
- **-Wsign-compare** — сравнение знакового и беззнакового
- **-Wsizeof-pointer-div** — подозрительное использование sizeof в операциях деления
- **-Wsizeof-pointer-memaccess** — параметры подозрительной длины (например, sizeof(ptr))
- **-Wstrict-aliasing** — использование кода нарушающего процесс оптимизации
- **-Wstrict-overflow** — недопустимое знаковое переполнение (например, x+1 > x)

Компилятор (предупреждения)

- **-Wswitch** — отсутствуют возможные case условия
- **-Wtautological-compare** — результат условия всегда определен (например, i>i)
- **-Wtrigraphs** — использование триграфов (??<) с возможностью двоякого смысла
- **-Wuninitialized** — возможно не инициализированная переменная
- **-Wunknown-pragmas** — использование непонятной #pragma директивы
- **-Wunused-function** — функция объявлена но не используется
- **-Wunused-label** — декларация объявлена но не используется
- **-Wunused-value** — вычисляемое значение не используется
- **-Wunused-variable** — переменная не используется
- **-Wvolatile-register-var** — переменная регистра объявлена изменчивой

Компилятор (доп.предупреждения, -Wextra)

- **-Wclobbered** — возможность изменения переменных при использовании longjmp или vfork
- **-Wcast-function-type** — указатель на функцию отображается на несовместную функцию
- **-Wempty-body** — пустое тело условия
- **-Wignored-qualifiers** — возвращаемый тип функции имеет классификатор типа const
- **-Wimplicit-fallthrough** — условие в switch переходит в другое условие (возможно пропущен break|return)
- **-Wmissing-field-initializers** — инициализатор структуры инициализирует не все поля
- **-Wmissing-parameter-type** — параметр функции без типа
- **-Wold-style-declaration** — устаревший формат деклараций
- **-Woverride-init** — переопределение инициализированного поля
- **-Wtype-limits** — сравнение всегда истинно или ложно из-за ограниченного диапазона данных
- **-Wuninitialized** — переменная не инициализирована
- **-Wshift-negative-value** — смещение отрицательного значения
- **-Wunused-parameter** — параметр функции не используется кроме объявления
- **-Wunused-but-set-parameter** — параметр функции назначается но не используется

Компилятор (остальное)

- **-Weffc++** must have опция. Не включается с помощью -Wextra или -Wall и содержит проверку рекомендаций Скотта Мейерса (http://www.aristeia.com/ddjpaper1.html):
  - Item 11: Define a copy constructor and an assignment operator for classes with dynamically allocated memory.
  - Item 12: Prefer initialization to assignment in constructors.
  - Item 14: Make destructors virtual in base classes.
  - Item 15: Have «operator=» return a reference to *this.
  - Item 23: Don't try to return a reference when you must return an object.
  - Item 6: Distinguish between prefix and postfix forms of increment and decrement operators.
  - Item 7: Never overload "&&", "||", or ",".
- **-Woverloaded-virtual** — перегрузка виртуальных фунций выглядит плохо.
- **-Wctor-dtor-privacy** — неиспользуемые классы — с приватным конструкторами и деструктором
- **-Wnon-virtual-dtor** — невиртуальный деструктор
- **-Wold-style-cast** — приведение в стиле C — это плохо
- **-Wconversion** -Wsign-conversion — ворнинг о преобразовании типа. при котором значение может измениться. Как ни странно, не входит в -Wall
- **-Wunreachable-code** — код, который никогда не будет выполнен

Среда программирования

- Динамическое отслеживание состояние кода
- Автоподстановка элементов/значений
- Подсветка/пояснение ошибочных конструкций
- Автозамена часто встречаемых ошибочных конструкций (пример: a.b на a->b)
- Оформление кода
- Отслеживание изменений

Специальные средства

- Более строгая формализация кода
- Набор тестов на наличие "здравого смысла" в коде
- Аналог работы компилятора с максимальными параметрами и минимальной производительностью
  - ***lint** Набор средств для разных языков программирования (splint)
  - **Coverity** Статический анализатор (http://www.coverity.com/)
  - **Cppcheck** Бесплатный анализатор на основе регулярных выражений (http://cppcheck.sourceforge.net/)
  - **Clang** Компилятор со встроенным статическим анализатором (http://clang.llvm.org/)
  - **Frama-C** Анализатор программ на языки Си (http://frama-c.com/)
  - **Goanna** Анализатор программ на языки Си/Си++ (http://www.redlizards.com/)
  - **PC-Lint** Весьма гибкий инструмент статического анализа (http://www.gimpel.com/)
  - ...

Splint

- Splint - инструмент для статической проверки программ на C для уязвимостей и ошибок программирования.
- Официальный сайт http://www.splint.org
- Проблемы, обнаруженные Splint, включают:
  - Разыменование нулевого указателя;
  - Использование возможно неопределенного хранилища или возвращаемого хранилища, которое не определено должным образом;
  - Несоответствия типов с большей точностью и гибкостью, чем предоставленные компиляторами C;
  - Нарушение информации скрывается;
  - Ошибки управления памятью, включая использование оборванных ссылок и утечек памяти;
  - Опасное сглаживание;
  - Модификации и использование глобальных переменных, которые не соответствуют указанным интерфейсам;
  - Проблемный поток управления, такой как вероятные бесконечные циклы, попадает в случаи или неполные swith и подозрительные утверждения;
  - Уязвимости переполнения буфера;
  - Опасные макрореализации или вызовы;
  - Нарушения индивидуальных соглашений об именах.

Splint

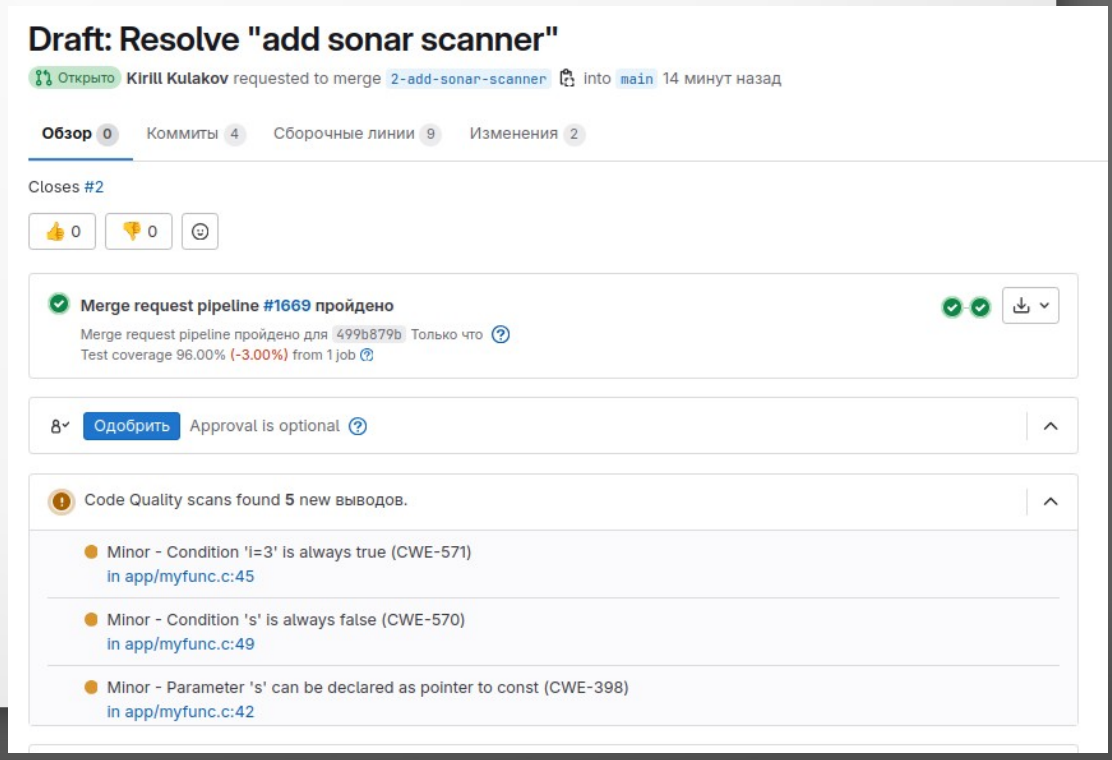
bool.c	Running Splint

# include "bool.h" 	
int f (int i, char *s,
bool b1, bool b2)
{
6 if (i = 3)
7 return b1;
8 if (!i || s)
9 return i;
10 if (s)
11 return 7;
12 if (b1 == b2)
13 return 3;
14 return 2;
}
	> splint bool.c +predboolptr –booltype bool
	
	bool.c:6: Test expression for if is assignment expression: i = 3
	bool.c:6: Test expression for if not bool, type int: i = 3
	bool.c:7: Return value type bool does not match declared type int: b1
	bool.c:8: Operand of ! is non-boolean (int): !i
	bool.c:8: Right operand of || is non-boolean (char *): !i || s
	bool.c:10: Test expression for if not bool, type char *: s
	bool.c:12: Use of == with bool variables (risks inconsistency because of multiple true values): b1 == b2
	
	Finished checking --- 7 code warnings found

CppCheck

- Проект с открытым исходным кодом (open source)
- https://github.com/danmar/cppcheck/
- Генерация отчета в формате XML
- Поддержка Gitlab code quality report формата

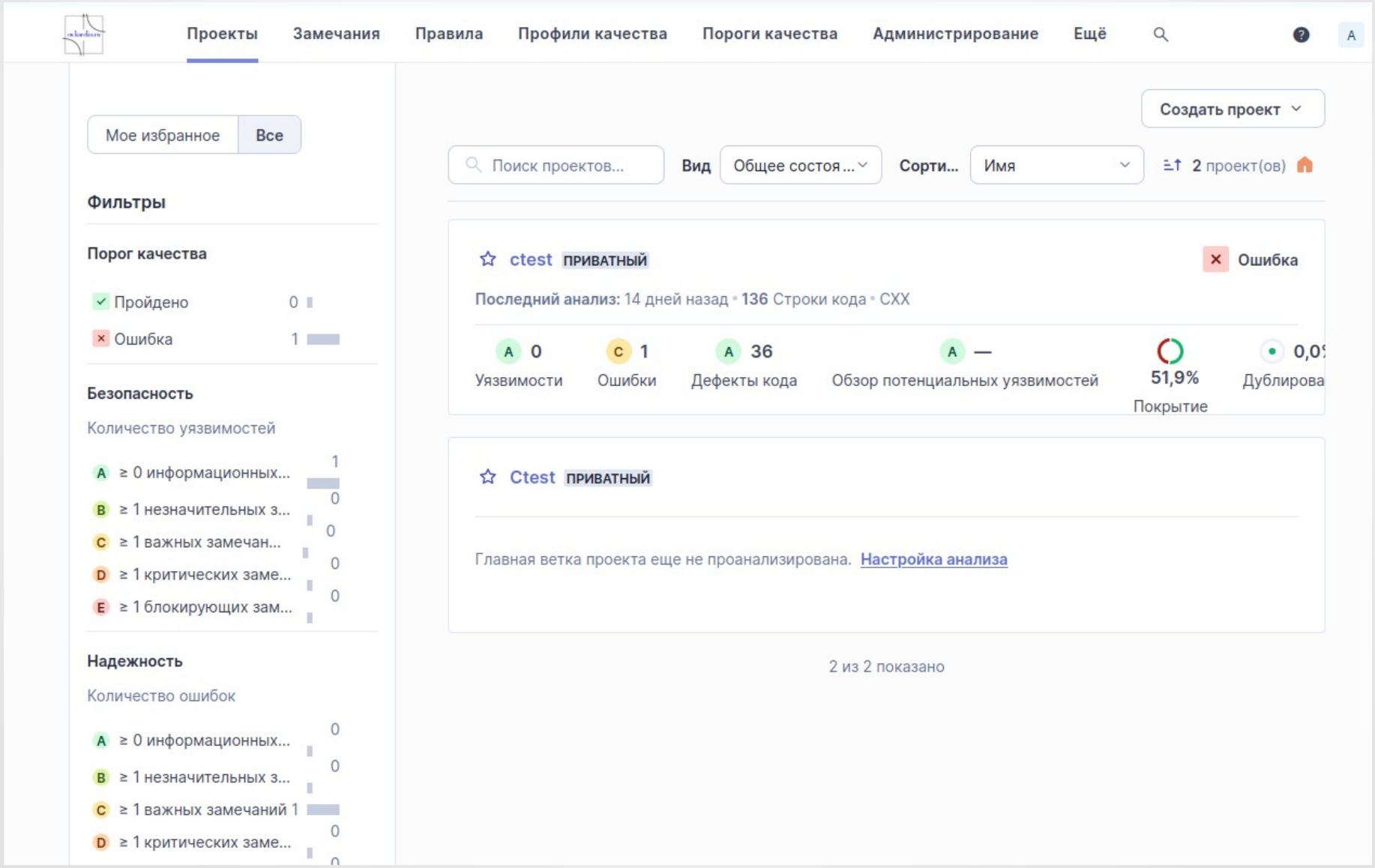
```yaml
cppcheck:
  stage: test
  script:
    - '( apt-get update -y && apt-get install cppcheck python3-pip -y )'
    - pip install cppcheck-codequality --break-system-packages
    # Generate CppCheck report as XML
    - cppcheck --xml --enable=all ./app 2> cppcheck.xml
    # Convert to a Code Climate JSON report
    - cppcheck-codequality --input-file cppcheck.xml --output-file cppcheck.json
  artifacts:
    reports:
      codequality: cppcheck.json
    paths:
      - cppcheck.xml

```



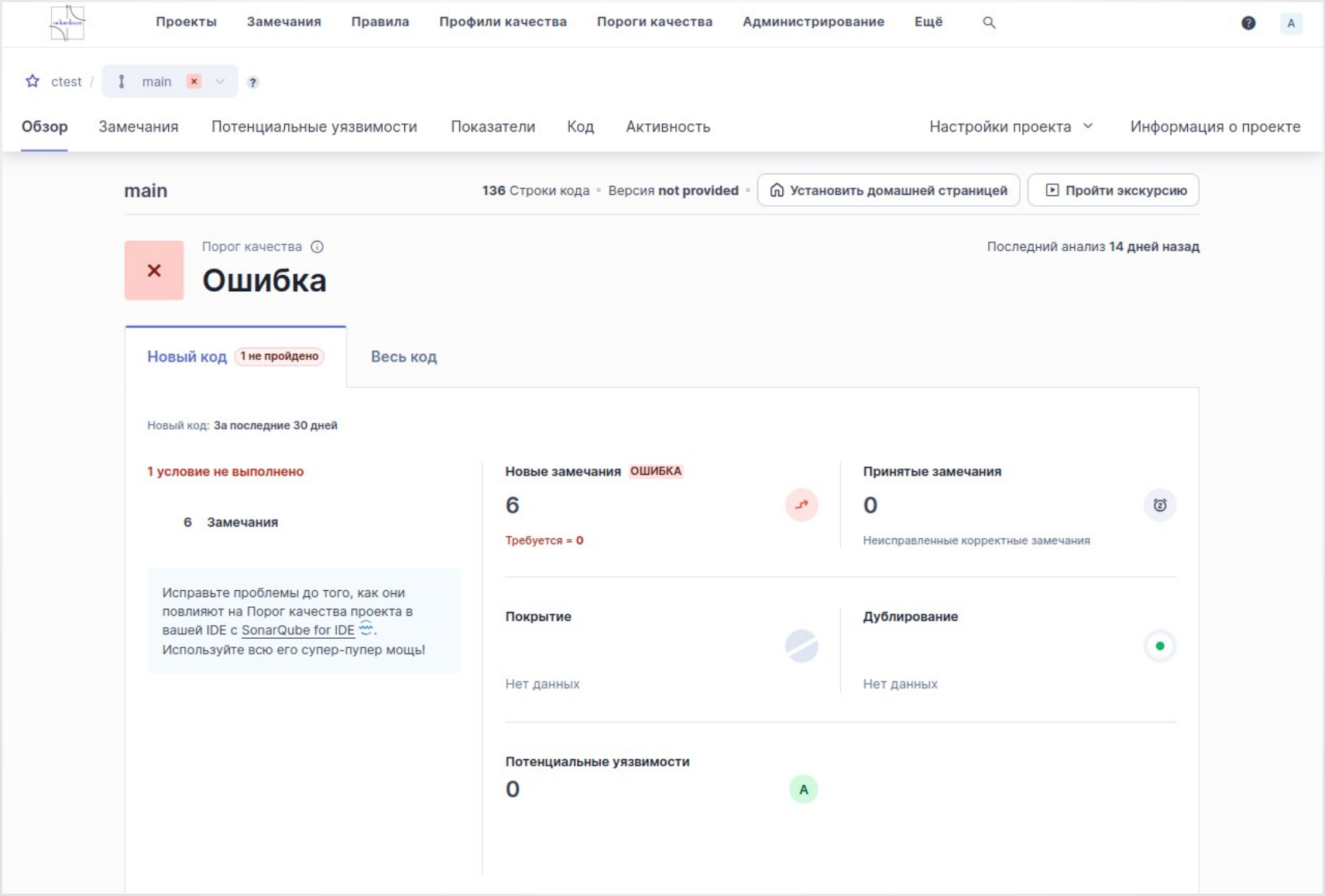
SonarQube

- Платформа для непрерывного анализа и измерения качества программного кода
- Использует встроенные анализаторы (платно) или результаты внешних статических анализаторов (бесплатно)
  - Используется SonarQube Community + sonar-cxx
- Правила для отображения содержания ошибок
- Профили качества для группирования правил
- Пороги качества для определения результата
- Интеграция с Gitlab
- Отображение результата анализа всего кода (gitlab отображает результат изменений в запросе на слияние)

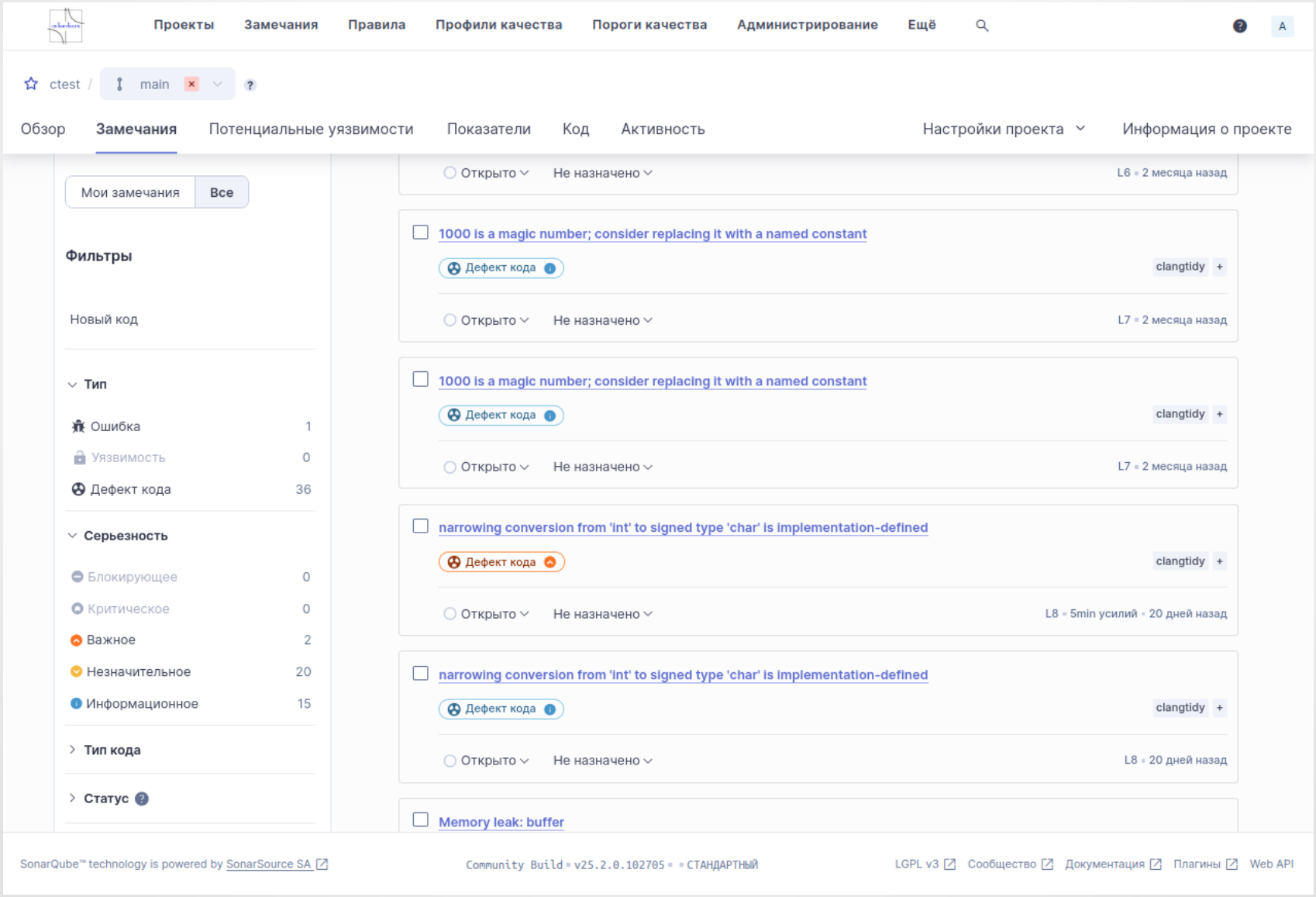
SonarQube


Проекты
Замечания
Правила
Профили качества
Пороги качества
Администрирование
Ещё
A
Мое избранное
Все
Фильтры
Порог качества
Пройдено 0
Ошибка 1
Безопасность
Количество уязвимостей
A ≥ 0 информационных... 1
B ≥ 1 незначительных з... 0
C ≥ 1 важных замечан... 0
D ≥ 1 критических заме... 0
E ≥ 1 блокирующих зам... 0
Надежность
Количество ошибок
A ≥ 0 информационных... 0
B ≥ 1 незначительных з... 0
C ≥ 1 важных замечаний 1
D ≥ 1 критических заме... 0
Создать проект
Поиск проектов...
Вид
Общее состоя...
Сорти...
Имя
2 проект(ов)
ctest ПРИВАТНЫЙ
Ошибка
Последний анализ: 14 дней назад • 136 Строки кода • CXX
A 0
Уязвимости
C 1
Ошибки
A 36
Дефекты кода
A —
Обзор потенциальных уязвимостей
51,9%
Покрытие
Дублирова
Ctest ПРИВАТНЫЙ
Главная ветка проекта еще не проанализирована. Настройка анализа
2 из 2 показано

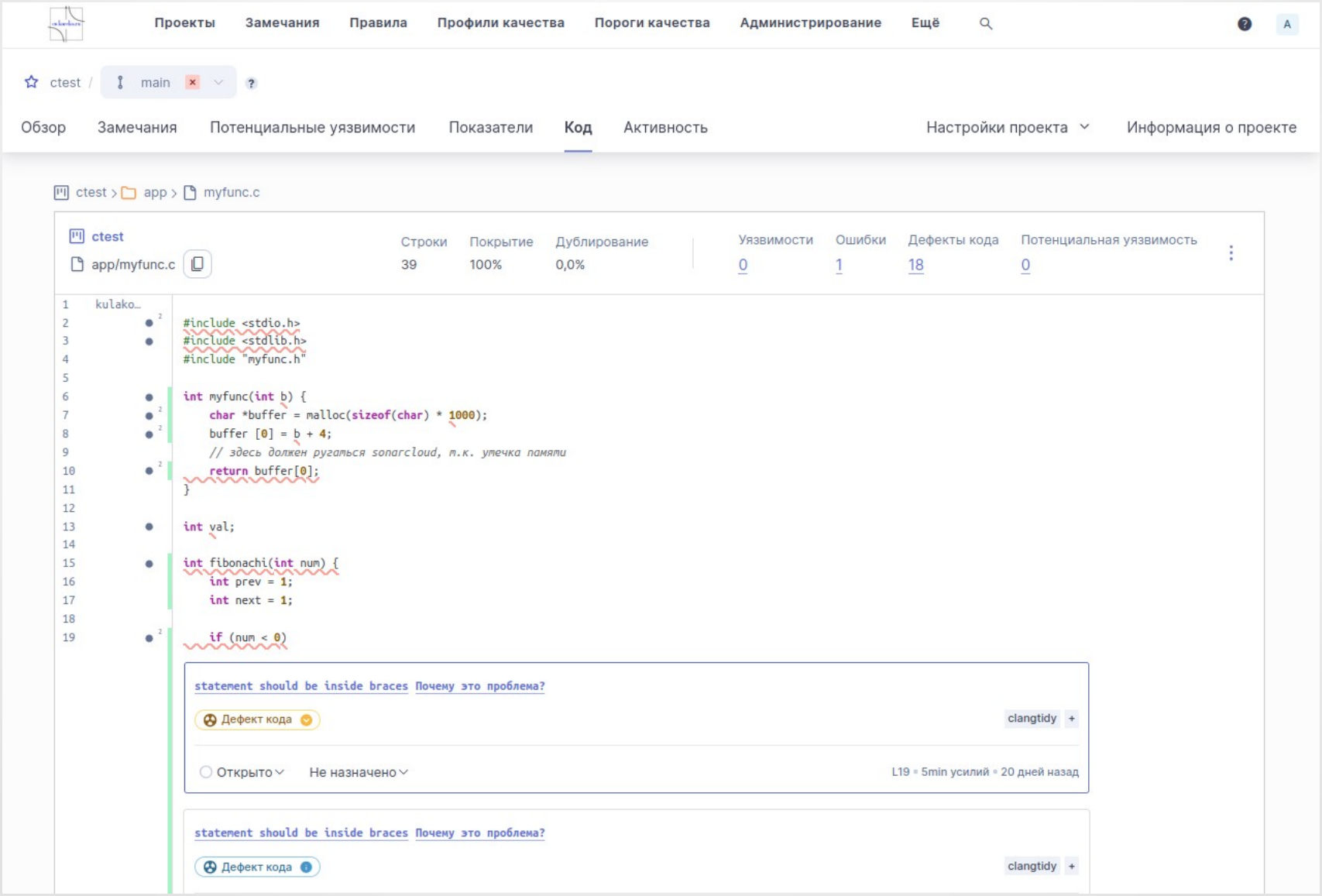
SonarQube


Проекты
Замечания
Правила
Профили качества
Пороги качества
Администрирование
Ещё
A
ctest / main
Обзор
Замечания
Потенциальные уязвимости
Показатели
Код
Активность
Настройки проекта
Информация о проекте
main
136 Строки кода • Версия not provided •
Установить домашней страницей
Пройти экскурсию
Порог качества
Ошибка
Последний анализ 14 дней назад
Новый код
1 не пройдено
Весь код
Новый код: За последние 30 дней
1 условие не выполнено
6 Замечания
Исправьте проблемы до того, как они повлияют на Порог качества проекта в вашей IDE с SonarQube for IDE.
Используйте всю его супер-пупер мощь!
Новые замечания
ОШИБКА
6
Требуется = 0
Принятые замечания
0
Неисправленные корректные замечания
Покрытие
Нет данных
Дублирование
Нет данных
Потенциальные уязвимости
0
A

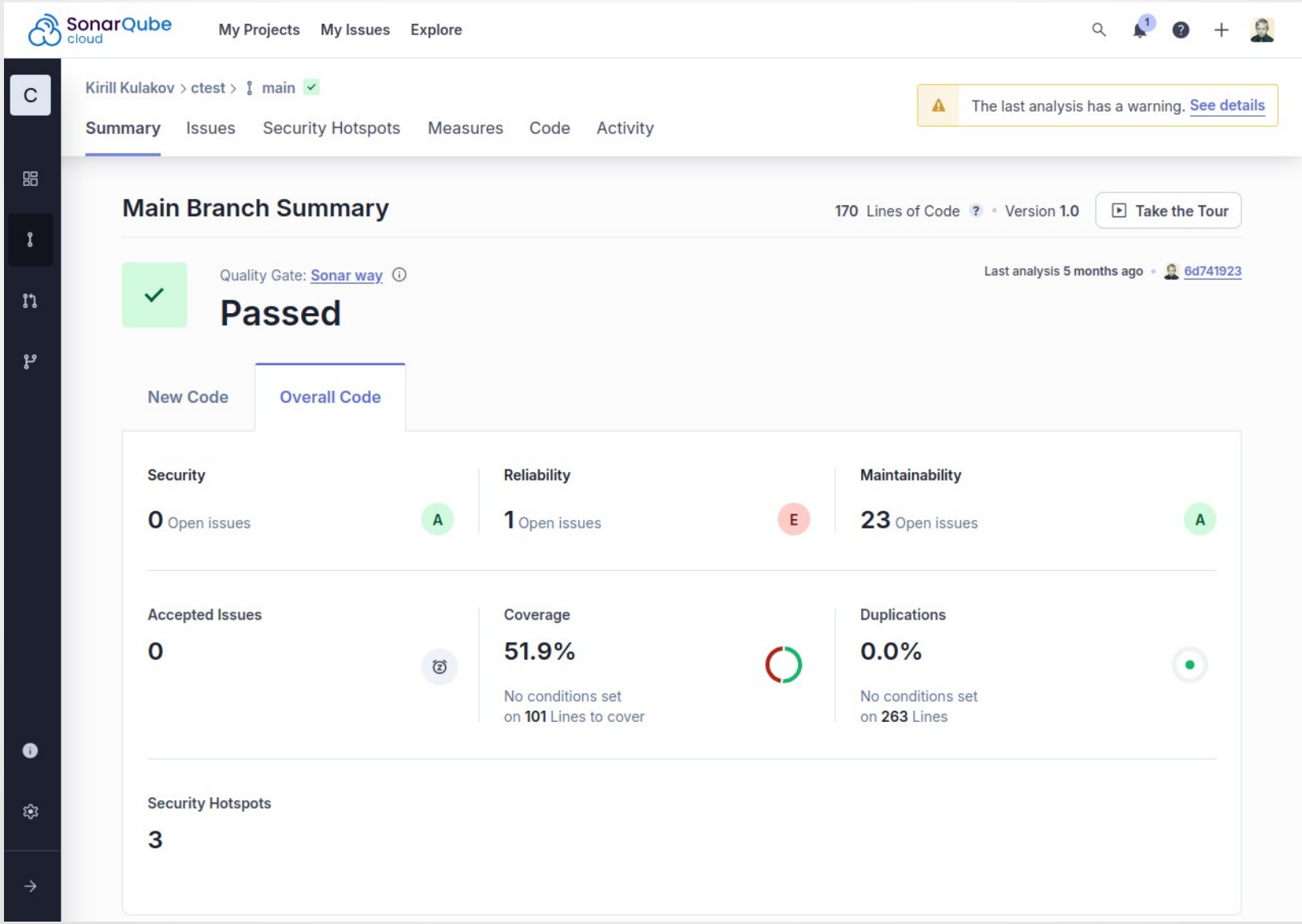
SonarQube


Проекты
Замечания
Правила
Профили качества
Пороги качества
Администрирование
Ещё
A
ctest / main
Обзор
Замечания
Потенциальные уязвимости
Показатели
Код
Активность
Настройки проекта
Информация о проекте
Мои замечания
Все
Фильтры
Новый код
Тип
Ошибка 1
Уязвимость 0
Дефект кода 36
Серьезность
Блокирующее 0
Критическое 0
Важное 2
Незначительное 20
Информационное 15
Тип кода
Статус
Открыто
Не назначено
L6 • 2 месяца назад
1000 is a magic number; consider replacing it with a named constant
Дефект кода
clangtidy
L7 • 2 месяца назад
narrowing conversion from 'int' to signed type 'char' is implementation-defined
L8 • 5min усилий • 20 дней назад
L8 • 20 дней назад
Memory leak: buffer
SonarQube™ technology is powered by SonarSource SA
Community Build • v25.2.0.102705 • • СТАНДАРТНЫЙ
LGPL v3
Сообщество
Документация
Плагины
Web API

SonarQube

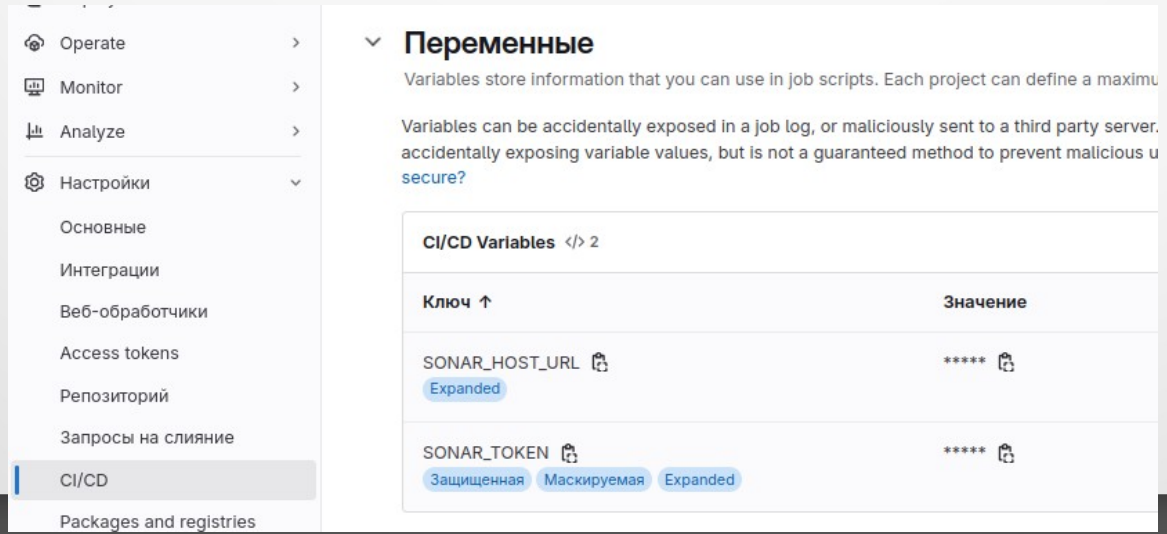

Проекты
Замечания
Правила
Профили качества
Пороги качества
Администрирование
Ещё
ctest / main
Обзор
Замечания
Потенциальные уязвимости
Показатели
Код
Активность
Настройки проекта
Информация о проекте
ctest > app > myfunc.c
ctest
app/myfunc.c
Строки 39
Покрытие 100%
Дублирование 0,0%
Уязвимости 0
Ошибки 1
Дефекты кода 18
Потенциальная уязвимость 0
#include <stdio.h>
#include <stdlib.h>
#include "myfunc.h"
int myfunc(int b) {
char *buffer = malloc(sizeof(char) * 1000);
buffer [0] = b + 4;
// здесь должен ругаться sonarcloud, т.к. утечка памяти
return buffer[0];
}
int val;
int fibonachi(int num) {
int prev = 1;
int next = 1;
if (num < 0)
statement should be inside braces Почему это проблема?
Дефект кода
clangtidy
Открыто
Не назначено
L19 • 5min усилий • 20 дней назад
statement should be inside braces Почему это проблема?
Дефект кода
clangtidy

SonarQube


SonarQube cloud
My Projects
My Issues
Explore
Kirill Kulakov > ctest > main
The last analysis has a warning. See details
Summary
Issues
Security Hotspots
Measures
Code
Activity
Main Branch Summary
170 Lines of Code
Version 1.0
Take the Tour
Quality Gate: Sonar way
Passed
Last analysis 5 months ago
6d741923
New Code
Overall Code
Security
0 Open issues
A
Reliability
1 Open issues
E
Maintainability
23 Open issues
A
Accepted Issues
0
Coverage
51.9%
No conditions set on 101 Lines to cover
Duplications
0.0%
No conditions set on 263 Lines
Security Hotspots
3

SonarQube

- Схема работы:
  - Настраиваем проект в SonarQube (импорт с Gitlab)
  - Добавляем конфигурационный файл sonar-project.properties в репозиторий
  - В gitlab CI/CD запускаем Clang-Tidy (пример)
  - Токен в настройки проекта
  - Делаем экспорт результата в SonarQube сервер



SonarQube

- Пример конфигурационного файла

sonar-project.properties 398 B

```
1  sonar.projectKey=ctest
2  sonar.qualitygate.wait=true
3
4  # mandatory parameter: files to be handled by the cxx plugin
5  sonar.cxx.file.suffixes=.cxx,.cpp,.cc,.c,.hxx,.hpp,.hh,.h
6
7  sonar.exclusions=coverage/**,coverage.xml,cppcheck.xml
8
9  sonar.cxx.cobertura.reportPaths=coverage.xml
10 sonar.cxx.cppcheck.reportPaths=cppcheck.xml
11 sonar.cxx.clangtidy.encoding=UTF-8
12 sonar.cxx.clangtidy.reportPaths=clang-tidy.txt
```

SonarQube

- Запуск Clang-Tidy в Gitlab CI/CD

- clang-tidy.sh

```
export CPP_SRC_FILES=$(find ./app -name "*.*" |
grep -E "(\.c$|\.cc$|\.cpp$|\.h$|\.hpp$)")


if [ -n "$CPP_SRC_FILES" ]; \
 then clang-tidy --quiet —checks='*,-llvmlibc-restrict-
system-libc-headers' \
 --header-filter=.*/app/.* $CPP_SRC_FILES \
 -- -Iusr/include > clang-tidy.txt; fi;
```

```
82
83 clang_tidy:
84   image:
85     name: xianpengshen/clang-tools:19
86   stage: test
87   script:
88     - ./clang-tidy.sh
89   artifacts:
90     paths:
91       - clang-tidy.txt
```

SonarQube

- CI/CD задача для экспорта

```
93  export_sonar:
94    stage: deploy
95
96    dependencies:
97      - build_modules
98      - cppcheck
99      - clang_tidy
100
101   image:
102     name: sonarsource/sonar-scanner-cli:11
103     entrypoint: [""]
104
105   variables:
106     SONAR_USER_HOME: "${CI_PROJECT_DIR}/.sonar"  # Defines the location of the analysis task cache
107     GIT_DEPTH: "0"  # Tells git to fetch all the branches of the project, required by the analysis task
108
109   cache:
110     policy: pull-push
111     key: "sonar-cache-$CI_COMMIT_REF_SLUG"
112     paths:
113       - "${SONAR_USER_HOME}/cache"
114       - sonar-scanner/
115
116   script:
117     - cat /builds/kulakov/ctest/clang-tidy.txt
118     - sonar-scanner -Dsonar.host.url="${SONAR_HOST_URL}"
119   allow_failure: true
120   rules:
121     - if: $CI_PIPELINE_SOURCE == 'merge_request_event'
122     - if: $CI_COMMIT_BRANCH == 'master'
123     - if: $CI_COMMIT_BRANCH == 'main'
124     - if: $CI_COMMIT_BRANCH == 'develop'
125
```

Valgrind

- Valgrind — инструментальное программное обеспечение, предназначенное для отладки использования памяти, обнаружения утечек памяти, а также профилирования
  - отслеживание работы с памятью (memcheck)
  - отслеживание вызовов (callgrind)
  - отслеживание работы с кэшем (cachegrind)
  - …
- Запуск:

  valgrind <app_file>

Valgrind (нет утечек)

```
kulakov@ultrabook:~/projects/testing/build-qmake-gtest-Desktop-Debug/app$ valgrind ./app
==31923== Memcheck, a memory error detector
==31923== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==31923== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==31923== Command: ./app
==31923==
Hello World!
==31923==
==31923== HEAP SUMMARY:
==31923==     in use at exit: 0 bytes in 0 blocks
==31923==   total heap usage: 4 allocs, 4 frees, 2,208 bytes allocated
==31923==
==31923== All heap blocks were freed -- no leaks are possible
==31923==
==31923== For counts of detected and suppressed errors, rerun with: -v
==31923== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Valgrind (утечки)

==32656== Memcheck, a memory error detector

==32656== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==32656== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==32656== Command: ./app

==32656==

Hello World!

==32656==

==32656== HEAP SUMMARY:

==32656== in use at exit: 1,000 bytes in 1 blocks

==32656== total heap usage: 5 allocs, 4 frees, 3,208 bytes allocated

==32656==

==32656== LEAK SUMMARY:

==32656== **definitely lost: 1,000 bytes in 1 blocks**

==32656== indirectly lost: 0 bytes in 0 blocks

==32656== possibly lost: 0 bytes in 0 blocks

==32656== still reachable: 0 bytes in 0 blocks

==32656== suppressed: 0 bytes in 0 blocks

==32656== Rerun with --leak-check=full to see details of leaked memory

==32656==

==32656== For counts of detected and suppressed errors, rerun with: -v

==32656== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

Valgrind (детали утечки)

kulakov@ultrabook:~/projects/testing/build-qmake-gtest-Desktop-Debug/app$ valgrind **--leak-check=full** ./app

==32759== Memcheck, a memory error detector

==32759== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.

==32759== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info

==32759== Command: ./app

==32759==

Hello World!

==32759==

==32759== HEAP SUMMARY:

==32759== in use at exit: 1,000 bytes in 1 blocks

==32759== total heap usage: 5 allocs, 4 frees, 3,208 bytes allocated

==32759==

==32759== 1,000 bytes in 1 blocks are definitely lost in loss record 1 of 1

==32759== at 0x4C2DB8F: malloc (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so)

==32759== by 0x400EA2: myfunc (myfunc.c:6)

==32759== by 0x400E63: main (main.c:8)

==32759==

...

Callgrind

- Запуск

```
kulakov@ultrabook:~/projects/testing/build-qmake-gtest-Desktop-Debug/app$ valgrind --tool=callgrind ./app
==446== Callgrind, a call-graph generating cache profiler
==446== Copyright (C) 2002-2015, and GNU GPL'd, by Josef Weidendorfer et al.
==446== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==446== Command: ./app
==446==
==446== For interactive control, run 'callgrind_control -h'.
Hello World!
==446==
==446== Events    : Ir
==446== Collected : 239342
==446==
==446== I   refs:      239,342
```

Callgrind

- Анализ

kulakov@ultrabook:~/projects/testing/build-qmake-gtest-Desktop-Debug/app$ **callgrind_annotate**

Reading data from 'callgrind.out.421'...

…….

```
--------------------------------------------------------------------------------
    Ir  file:function
--------------------------------------------------------------------------------
56,475  /build/glibc-Cl5G7W/glibc-2.23/elf/dl-addr.c:_dl_addr [/lib/x86_64-linux-gnu/libc-2.23.so]
33,910  /build/glibc-Cl5G7W/glibc-2.23/elf/dl-lookup.c:do_lookup_x [/lib/x86_64-linux-gnu/ld-2.23.so]
21,802  /build/glibc-Cl5G7W/glibc-2.23/elf/dl-lookup.c:_dl_lookup_symbol_x [/lib/x86_64-linux-gnu/ld-2.23.so]
20,616  ???:gcov_do_dump [/home/kulakov/projects/testing/build-qmake-gtest-Desktop-Debug/app/app]
17,553  /build/glibc-Cl5G7W/glibc-2.23/elf/../sysdeps/x86_64/dl-machine.h:_dl_relocate_object
```