Property-testing all the things in SerenityOS

Martin Janiczek
@janiczek

Property-testing all* the things in SerenityOS

Martin Janiczek
@janiczek

I love PBT!

I love PBT!?

Property Based Testing

```
unit_test "list reversing" {
  input = [1,2,3]
  reversed = reverse(input)
  assert(reversed == [3,2,1])
}
```

```
randomized_test "list reversing" (input: List[Int]) {
  reversed = reverse(input)
  twice = reverse(reversed)
  assert(twice == input)
}
```

Property Based Testing

Test with many random inputs

Property Based Testing

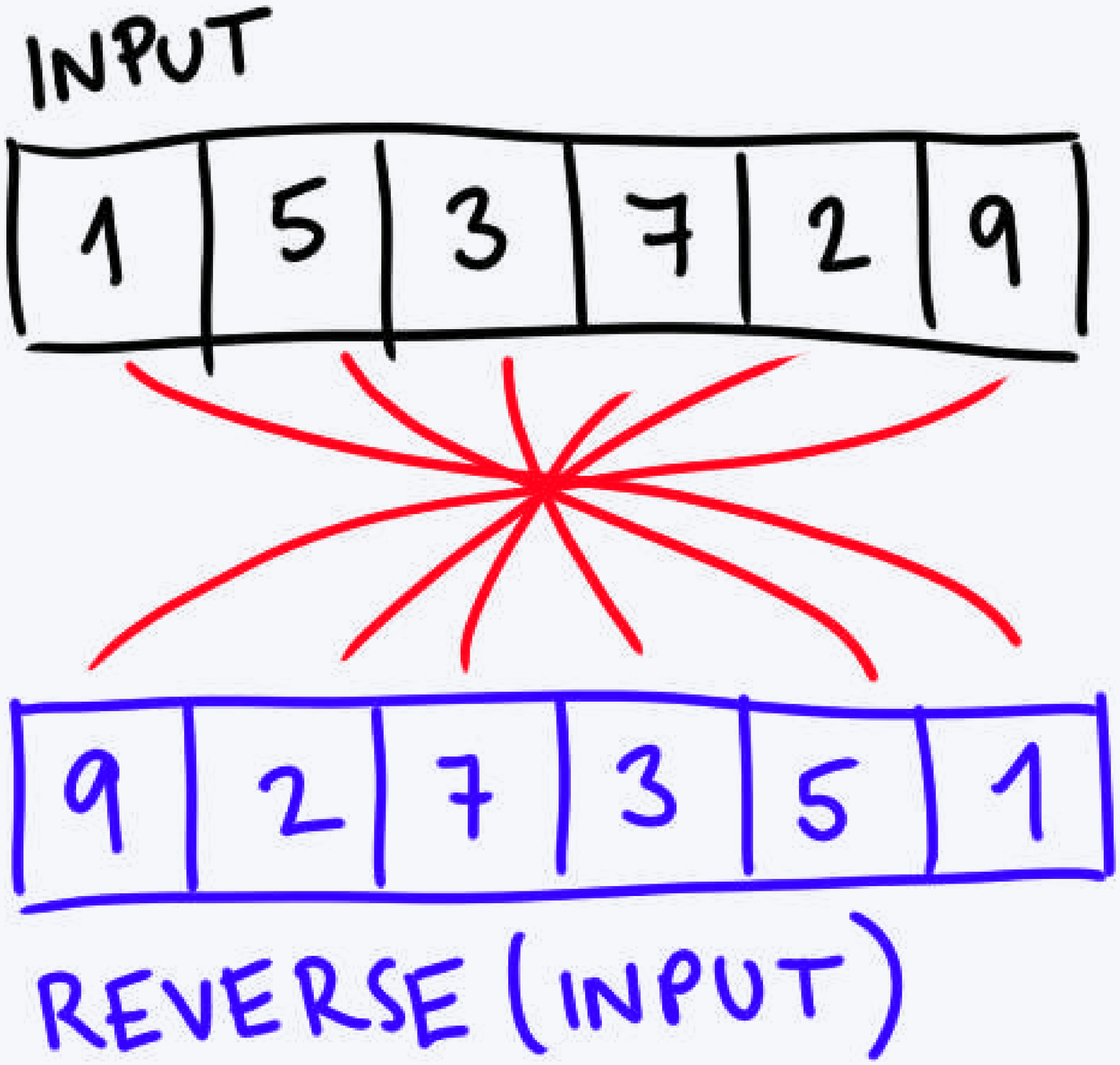
Test with many random inputs
Finds a minimal example of a failure

Property Based Testing

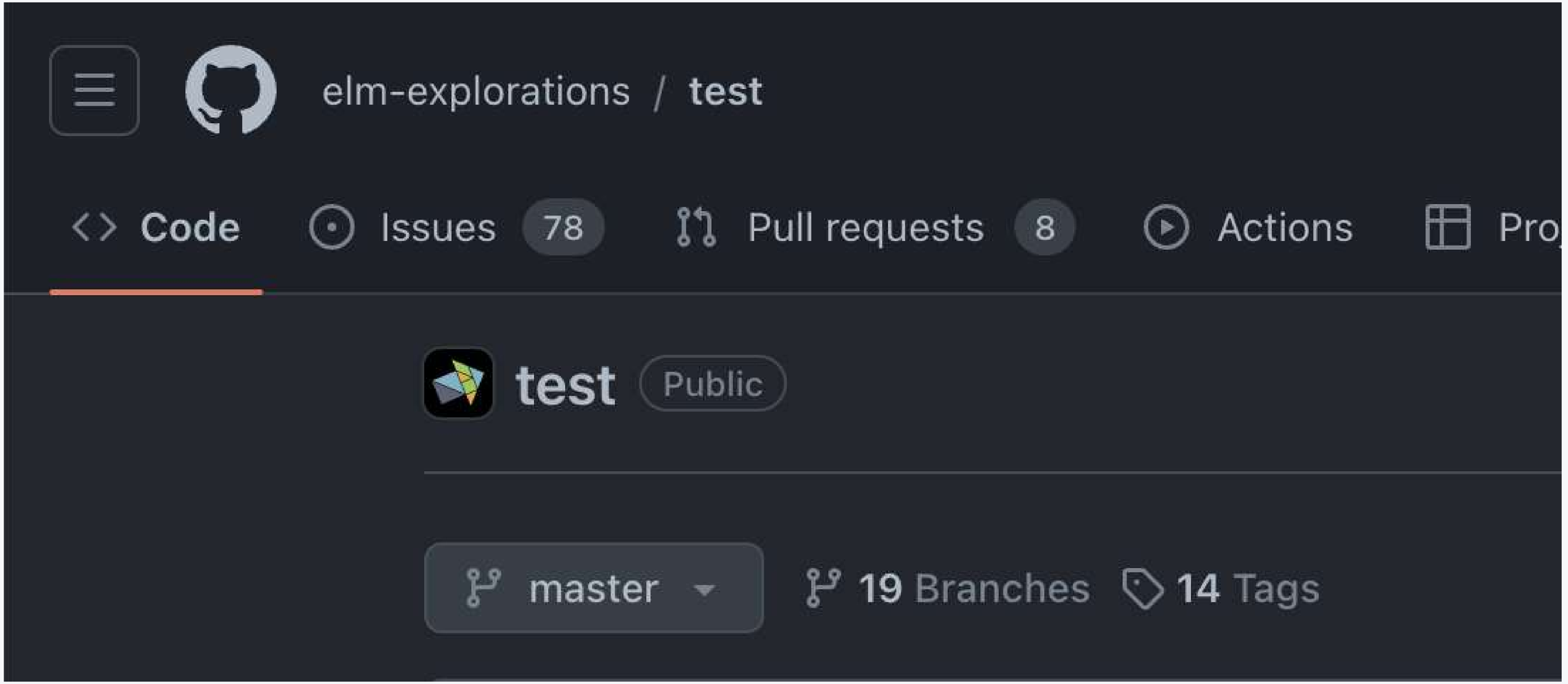
Test with many random inputs
Finds a minimal example of a failure
Focus on specification, not specific examples

```
randomized_test "list reversing" (input: List[Int]) {
  reversed = reverse(input)
  assert(reversed == ???)
}
```

```
randomized_test "list reversing" (input: List[Int]) {
  reversed = reverse(input)
  twice = reverse(reversed)
  assert(twice == input)
}
```


INPUT
1
5
3
7
2
9
9
2
7
3
5
1
REVERSE (INPUT)

I love PBT!


elm-explorations / test
Code
Issues 78
Pull requests 8
Actions
Pro
test
Public
master
19 Branches
14 Tags

Uncommon Fuzzers

custom : Generator a -> Shrinker a -> Fuzzer a

Build a custom `Fuzzer a` by providing a `Generator a` and a `Shrinker a`. Generators are defined in `elm/random`. Shrinkers are defined in the `Shrink` module. It is not possible to extract the generator and shrinker from an existing fuzzer.

This function should be considered for advanced uses. It's often easier to use `map` and other functions in this module to create a fuzzer.

Here is an example for a record:

```
import Random
import Shrink

type alias Position =
    { x : Int, y : Int }

position : Fuzzer Position
position =
    Fuzz.custom
        (Random.map2 Position (Random.int -100 100) (Rando
        (\{ x, y } -> Shrink.map Position (Shrink.int x)
```

Approaches

Approaches

1. QuickCheck doesn't keep constraints

Approaches

1. QuickCheck — doesn't keep constraints
2. Hedgehog — suffers when monadic bind is used

Approaches

1. QuickCheck doesn't keep constraints
2. Hedgehog suffers when monadic bind is used
3. Hypothesis ...actually pretty awesome?

Changes in 2.0.0

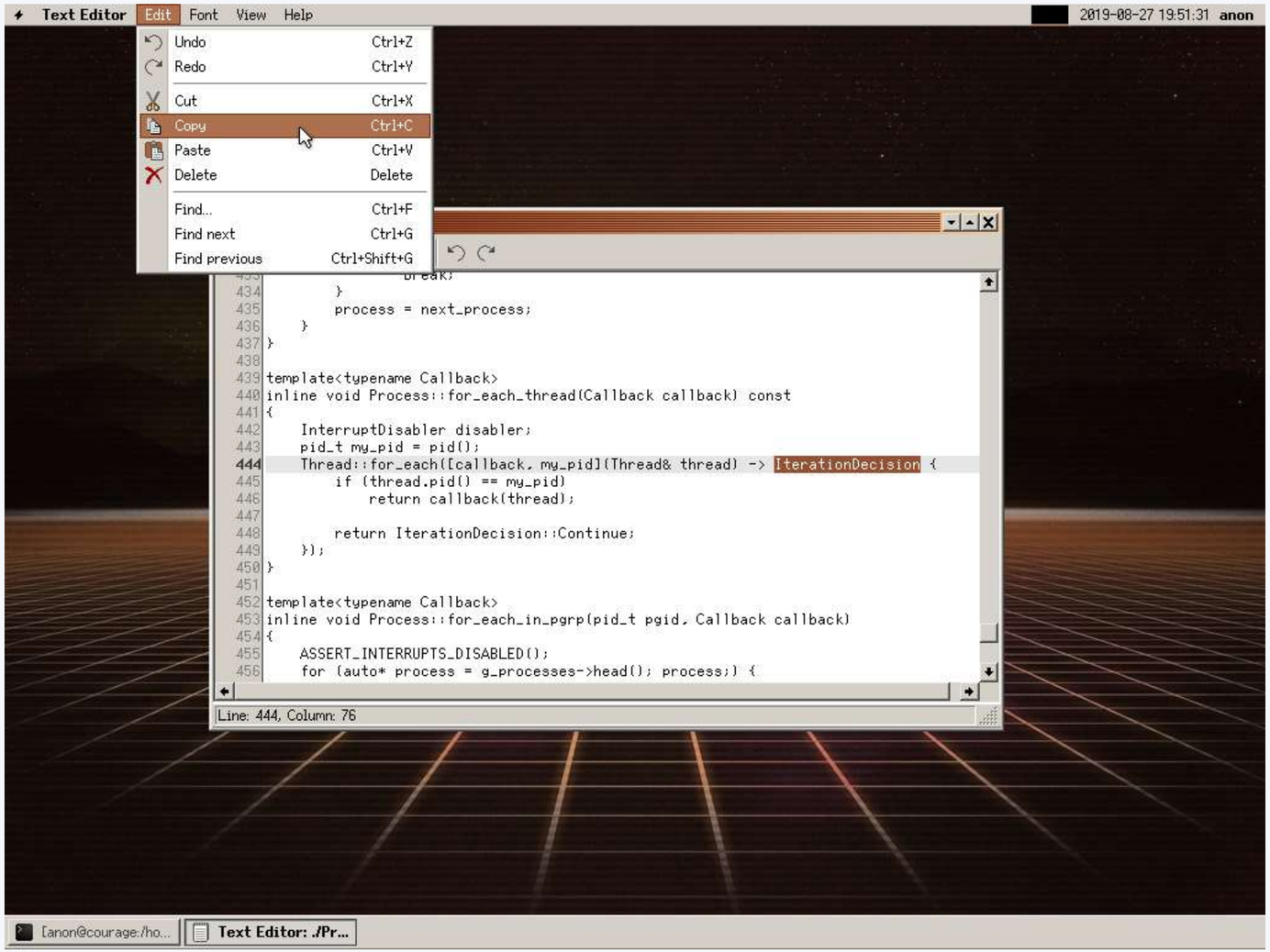
1. Fuzzing and shrinking reimplementation

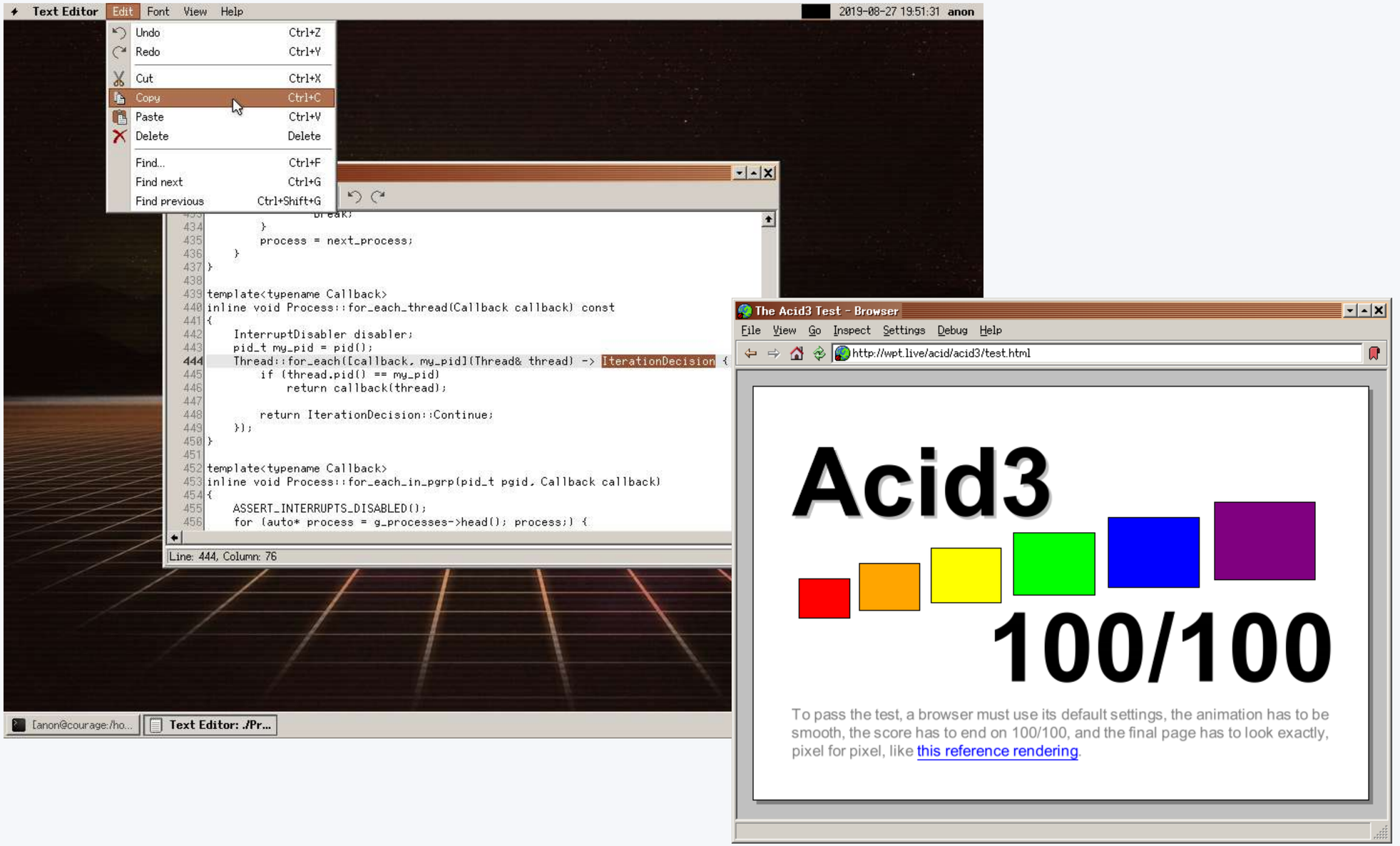
Fuzzing and shrinking has been reimplemented: the rose tree approach has been replaced with the "internal shrinking" approach found in the Python test library Hypothesis.

In short, shrinking is now done on the PRNG history instead of on the generated values themselves. This is hidden from the user: the `Shrink` module has now been removed.

This new approach allows us to reintroduce `Fuzz.andThen` and remove `Fuzz.custom`: in case you were forced to use `Fuzz.custom` and a `Random` generator, you'll now be able to express this logic with `Fuzz` alone.

I admire SerenityOS!


Text Editor
Edit
Font
View
Help
2019-08-27 19:51:31 anon
Undo Ctrl+Z
Redo Ctrl+Y
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Find... Ctrl+F
Find next Ctrl+G
Find previous Ctrl+Shift+G
break;
}
process = next_process;
}
}
template<typename Callback>
inline void Process::for_each_thread(Callback callback) const
{
InterruptDisabler disabler;
pid_t my_pid = pid();
Thread::for_each([callback, my_pid](Thread& thread) -> IterationDecision {
if (thread.pid() == my_pid)
return callback(thread);
return IterationDecision::Continue;
});
}
template<typename Callback>
inline void Process::for_each_in_pgrp(pid_t pgid, Callback callback)
{
ASSERT_INTERRUPTS_DISABLED();
for (auto* process = g_processes->head(); process;) {
Line: 444, Column: 76
[anon@courage:/ho...
Text Editor: ./Pr...


Text Editor
Edit
Font
View
Help
2019-08-27 19:51:31 anon
Undo Ctrl+Z
Redo Ctrl+Y
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Find... Ctrl+F
Find next Ctrl+G
Find previous Ctrl+Shift+G
}
process = next_process;
}
}
template<typename Callback>
inline void Process::for_each_thread(Callback callback) const
{
InterruptDisabler disabler;
pid_t my_pid = pid();
Thread::for_each([callback, my_pid](Thread& thread) -> IterationDecision {
if (thread.pid() == my_pid)
return callback(thread);
return IterationDecision::Continue;
});
}
template<typename Callback>
inline void Process::for_each_in_pgrp(pid_t pgid, Callback callback)
{
ASSERT_INTERRUPTS_DISABLED();
for (auto* process = g_processes->head(); process;) {
Line: 444, Column: 76
[anon@courage:/ho...
Text Editor: ./Pr...
The Acid3 Test - Browser
File View Go Inspect Settings Debug Help
http://wpt.live/acid/acid3/test.html
Acid3
100/100
To pass the test, a browser must use its default settings, the animation has to be smooth, the score has to end on 100/100, and the final page has to look exactly, pixel for pixel, like this reference rendering.

I love PBT!

I love PBT!

I admire SerenityOS!

I love PBT!

I admire SerenityOS!

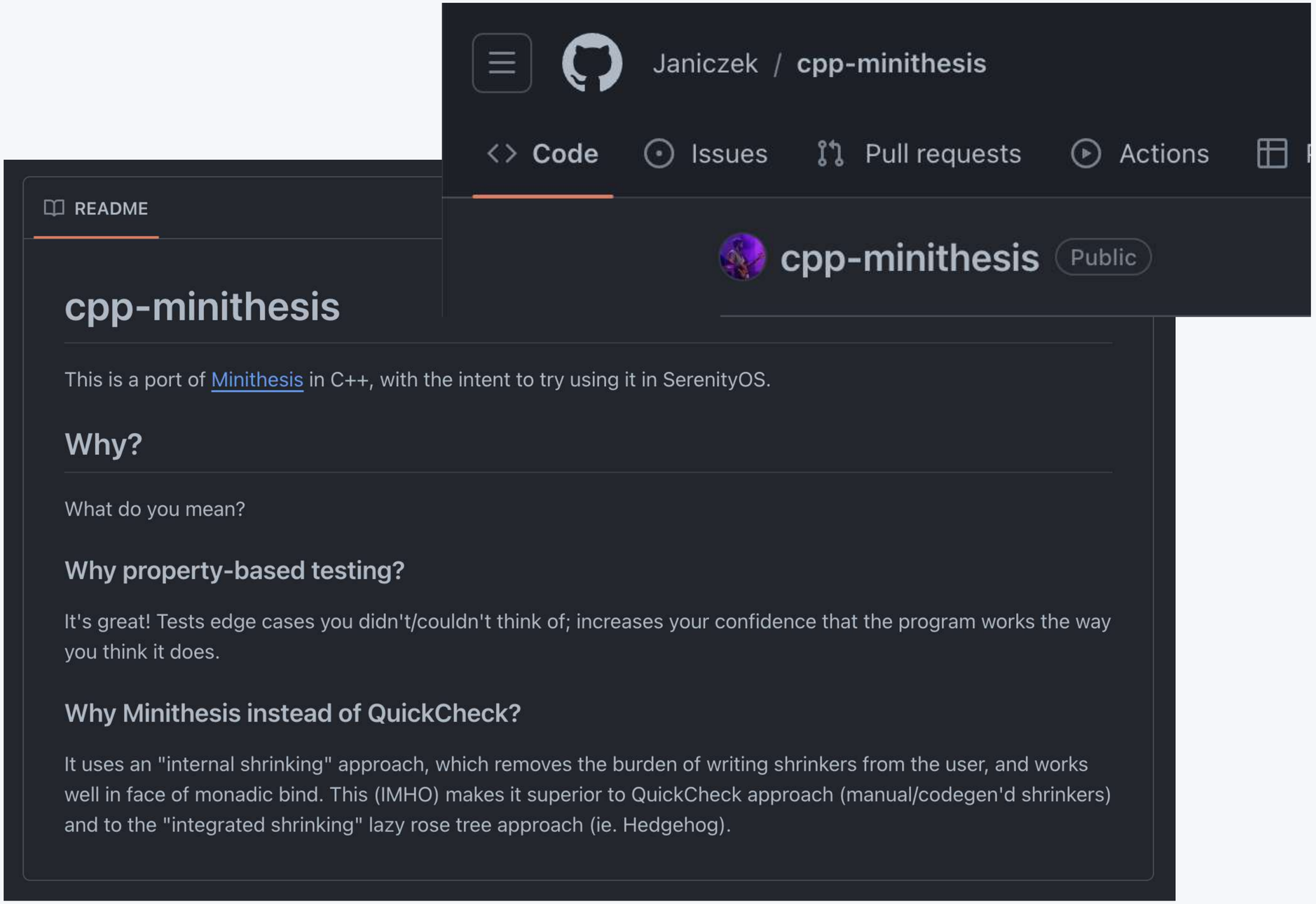

TAKE
THIGH

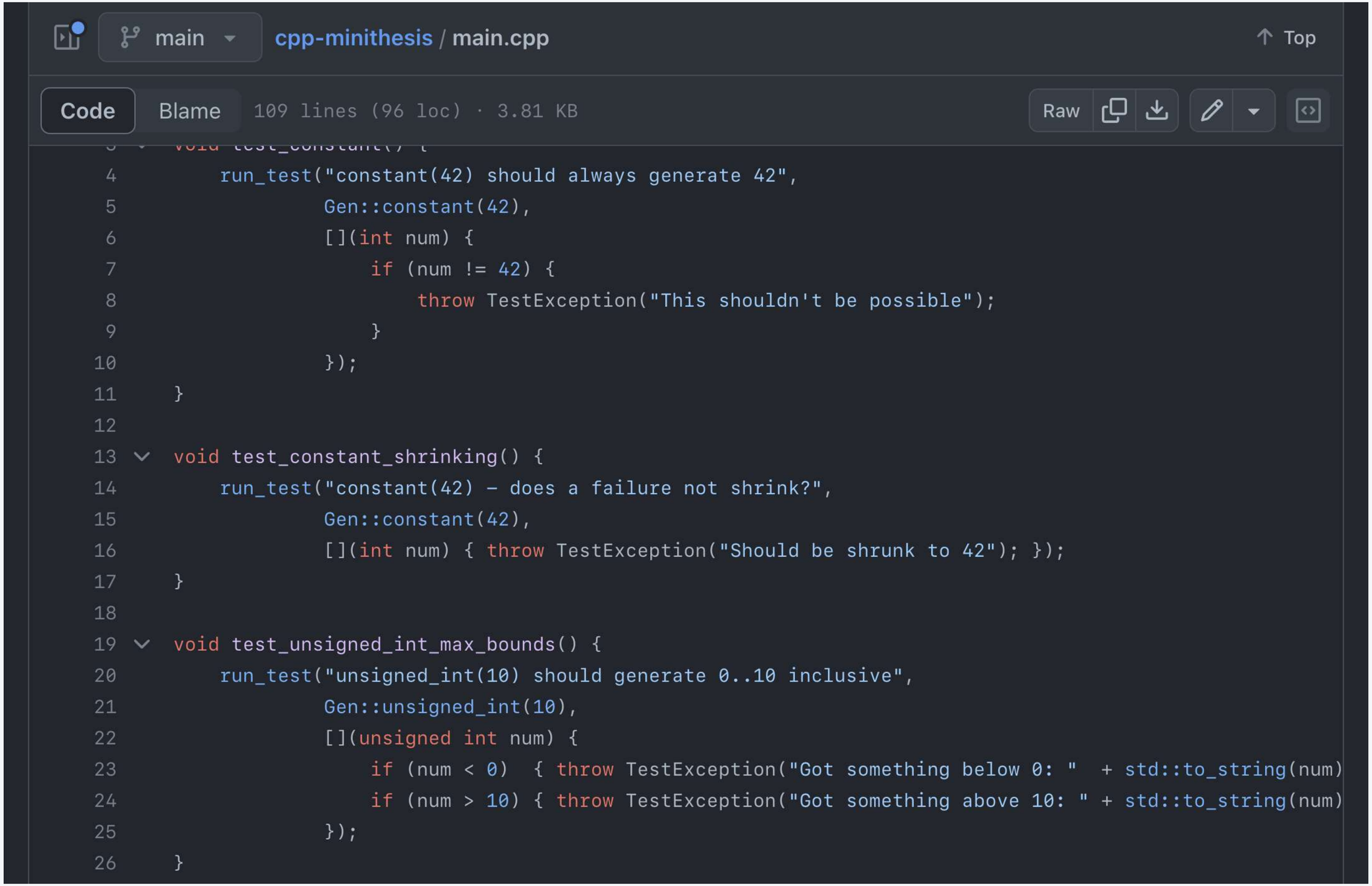

TAKE

August 24, 2023

12:53 PM **janiczek** Hey there 👋 I wonder, would there be interest in having property-based tests for the various SerenityOS libraries and apps? I've checked TestCase and it knows about unit tests and benchmark tests... but not PBT ones. I've also seen some fuzzing via LLVM.

5:37 PM **awesomekling** hello there! I'd say we're always interested any kind of testing that can surface unknown issues :)

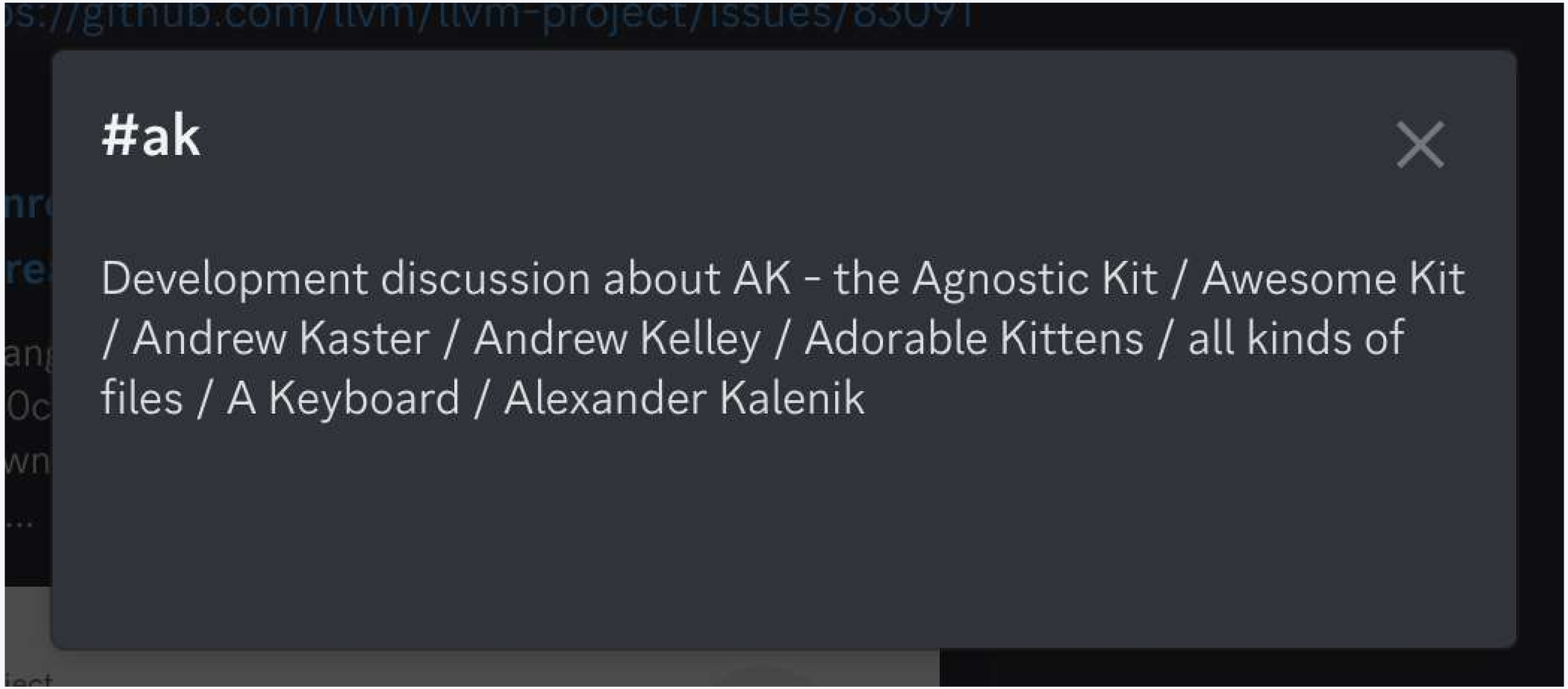

Janiczek / cpp-minithesis
Code
Issues
Pull requests
Actions
README
cpp-minithesis
Public
cpp-minithesis
This is a port of Minithesis in C++, with the intent to try using it in SerenityOS.
Why?
What do you mean?
Why property-based testing?
It's great! Tests edge cases you didn't/couldn't think of; increases your confidence that the program works the way you think it does.
Why Minithesis instead of QuickCheck?
It uses an "internal shrinking" approach, which removes the burden of writing shrinkers from the user, and works well in face of monadic bind. This (IMHO) makes it superior to QuickCheck approach (manual/codegen'd shrinkers) and to the "integrated shrinking" lazy rose tree approach (ie. Hedgehog).




```
    run_test("constant(42) should always generate 42",
             Gen::constant(42),
             [](int num) {
                 if (num != 42) {
                     throw TestException("This shouldn't be possible");
                 }
             });
}

void test_constant_shrinking() {
    run_test("constant(42) - does a failure not shrink?",
             Gen::constant(42),
             [](int num) { throw TestException("Should be shrunk to 42"); });
}

void test_unsigned_int_max_bounds() {
    run_test("unsigned_int(10) should generate 0..10 inclusive",
             Gen::unsigned_int(10),
             [](unsigned int num) {
                 if (num < 0)  { throw TestException("Got something below 0: "  + std::to_string(num)
                 if (num > 10) { throw TestException("Got something above 10: " + std::to_string(num)
             });
}
```

cpp-minithesis
STL

SerenityOS
AK


#ak
Development discussion about AK - the Agnostic Kit / Awesome Kit / Andrew Kaster / Andrew Kelley / Adorable Kittens / all kinds of files / A Keyboard / Alexander Kalenik

cpp-minithesis
STL

SerenityOS
AK

```elm
Test.property "bind"
  -- generation phase:
  (Gen.int 1 9
    |> Gen.andThen (\n1 ->
      Gen.int (n1 * 10) (n1 * 100)
        |> Gen.map (\n2 -> (n1, n2))
    )
  )
  -- testing phase:
  (\(n1, n2) ->
    Expect.all
      [ n1 >= 1 && n1 <= 9
      , n2 >= 10 && n2 <= 900
      ]
  )
```

```
RANDOMIZED_TEST_CASE(
  bind,
  // generation phase:
  Gen::number_u64(1,9).bind([](u64 n1) {
    return Gen::number_u64(n1 * 10, n1 * 100)
      .map([=](u64 n2){
        return Tuple<u64,u64> {n1, n2};
      });
  }),
  input
) {
  // testing phase:
  u64 n1 = input.get<0>();
  u64 n2 = input.get<1>();
  EXPECT(n1 ⩾ 1 && n1 ⩽ 9);
  EXPECT(n2 ⩾ 10 && n2 ⩽ 900);
}
```

```
Test.property "bind"
  -- generation phase:
  (Gen.int 1 9
    ▷ Gen.andThen (\n1 →
      Gen.int (n1 * 10) (n1 * 100)
        ▷ Gen.map (\n2 → (n1, n2))
    )
  )
  -- testing phase:
  (\(n1, n2) →
    Expect.all
      [ n1 ⩾ 1 && n1 ⩽ 9
      , n2 ⩾ 10 && n2 ⩽ 900
      ]
  )
```

```
RANDOMIZED_TEST_CASE(
  bind,
  // generation phase:
  Gen::number_u64(1,9).bind([](u64 n1) {
    return Gen::number_u64(n1 * 10, n1 * 100)
      .map([=](u64 n2){
        return Tuple<u64,u64> {n1, n2};
      });
  }),
  input
) {
  // testing phase:
  u64 n1 = input.get<0>();
  u64 n2 = input.get<1>();
  EXPECT(n1 ⩾ 1 && n1 ⩽ 9);
  EXPECT(n2 ⩾ 10 && n2 ⩽ 900);
}
```

```
RANDOMIZED_TEST_CASE(bind_like)
{
  GEN(n1, Gen::number_u64(1, 9));
  // n1 is just an int!
  EXPECT(n1 ⩾ 1 && n1 ⩽ 9);
  // feel free to generate again!
  GEN(n2, Gen::number_u64(n1 * 10, n1 * 100));
  EXPECT(n2 ⩾ 10 && n2 ⩽ 900);
}
```

```
// Values: fine!
Gen::oneOf(1, 5, 9)

// Functions: type inference sucks
Gen::oneOf([](){return Gen::number_u64(1,9);},
           [](){return Gen::number_u64(2,3);})
```

12:15 PM CxByte @janiczek

What you want is possible, but way too complex
You'd have to zip all args and CommonType each, produce a pack again, then splat it into Function along with the CommonType of the return types.

Instead take the type of the first one and force everything to be the same:

```
template <typename Fn, typename... Fns,
typename R = decltype(declval<F>())>
R one_of(Fn f, Fns... fns)
{
    Vector<Function<F>> ...;
    ...
}
```

LibTest: Add support for randomized tests #21191

Merged ADKaster merged 13 commits into `SerenityOS:master` from `Janiczek:property-based-tests` on Oct 27, 2023

Conversation 112 · Commits 13 · Checks 13 · Files changed 16

Janiczek commented on Sep 22, 2023 • edited · Contributor

Add a way to run randomized tests (commonly called "property-based"): that is, tests that generate random data to run the test case with, and if they find a failure they shrink the input to a minimal failing example before reporting it to the user.

See README.md for more in-depth description.

Examples:

```
// Tests/LibCompress/TestBrib.cpp
// This test didn't find anything but shows off a very common property we can test with this.
RANDOMIZED_TEST_CASE(roundtrip)
{
    GEN(buffer, Gen::vector(24KiB,[]{return (u8)Gen::unsigned_int(256);}));
    auto const compressed = MUST(Compress::GzipCompressor::compress_all(buffer));
    auto const decompressed = MUST(Compress::GzipDecompressor::decompress_all(compressed));
    EXPECT(buffer == decompressed);
}
```

```
// Tests/LibIMAP/TestQuotedPrintable.cpp
// The randomized test below found a deviation from the spec!

TEST_CASE(section_6_7_3_white_space_regressions)
{
    // Found by the randomized test below.

    // Throws the encoded tab/space at the end of the string away
    DECODE_EQUAL(""\t"ex,  ""ex);
    DECODE_EQUAL(" ex,  ""ex);

    // Doesn't throw the encoded tab/space in the middle of the string away
    DECODE_EQUAL(""\t"ex,  ""\t""ex);
    DECODE_EQUAL(" ex,  " ex);
}

RANDOMIZED_TEST_CASE(section_6_7_3_white_space)
{
    /* https://datatracker.ietf.org/doc/html/rfc2049#section-6.7

       (3) (White Space) Octets with values of 9 and 32 MAY be
           represented as US-ASCII TAB (HT) and SPACE characters,
           respectively, but MUST NOT be so represented at the end
           of an encoded line.  Any TAB (HT) or SPACE characters
           on an encoded line MUST thus be followed on that line
           by a printable character.  In particular, an "=" at the
           end of an encoded line, indicating a soft line break
           (see rule #5) may follow one or more TAB (HT) or SPACE
           characters.  It follows that an octet with decimal
           value 9 or 32 appearing at the end of an encoded line
           must be represented according to Rule #1.  This rule is
           necessary because some MTAs (Message Transport Agents,
           programs which transport messages from one user to
           another, or perform a portion of such transfers) are
           known to pad lines of text with SPACEs, and others are
           known to remove "white space" characters from the end
           of a line.  Therefore, when decoding a Quoted-Printable
           body, any trailing white space on a line must be
           deleted, as it will necessarily have been added by
           intermediate transport agents.
     */

    GEN(prefix, literals_gen());
    auto prefix_sv = vector_to_string_view(prefix);

    // Throws the encoded tab at the end of the string away
    StringBuilder tab_at_end;
    tab_at_end.append(prefix_sv);
    tab_at_end.append('\t');
    DECODE_EQUAL(tab_at_end.string_view(), prefix_sv);

    // Throws the encoded space at the end of the string away
    StringBuilder space_at_end;
    space_at_end.append(prefix_sv);
    space_at_end.append(' ');
    DECODE_EQUAL(space_at_end.string_view(), prefix_sv);

    GEN(suffix, literals_gen());
    auto suffix_sv = vector_to_string_view(suffix);

    // Doesn't throw the encoded tab in the middle of the string away
    StringBuilder tab_in_middle;
    tab_in_middle.append(prefix_sv);
    tab_in_middle.append('\t');
    tab_in_middle.append(suffix_sv);
    StringView tab_in_middle_sv = tab_in_middle.string_view();
    DECODE_EQUAL(tab_in_middle_sv, tab_in_middle_sv);

    // Doesn't throw the encoded space in the middle of the string away
    StringBuilder space_in_middle;
    space_in_middle.append(prefix_sv);
    space_in_middle.append(' ');
    space_in_middle.append(suffix_sv);
    StringView space_in_middle_sv = space_in_middle.string_view();
    DECODE_EQUAL(space_in_middle_sv, space_in_middle_sv);
}
```

(I'll make another PR with usages of `RANDOMIZED_TEST_CASE` that will build on this one; I appreciate this PR is already big as it is!)

> **Note**
>
> This being my first SerenityOS contribution, I'll be glad for any suggestions (code style, C++ tricks, namespace organization)!

github-actions bot added the `pr-needs-review` label on Sep 22, 2023

BuggieBot commented on Sep 22, 2023 · Member

Hello!

One or more of the commit messages in this PR do not match the SerenityOS code submission policy, please check the `Lint_commits` CI job for more details on which commits were flagged and why.
Please do not close this PR and open another, instead modify your commit message(s) with git commit --amend and force push those changes to update this PR.

Janiczek force-pushed the `property-based-tests` branch from `3b76be5` to `11a3b3c` 5 months ago · Compare

ADKaster commented on Sep 23, 2023 · Member

Before taking a real look at this, a few general comments from a scroll through:

- Make sure to fixup-squash any updates, or PR review comment changes.
- As written, this is one huge commit.
  - Can you split it into "atomic commits", that each change one thing, and build on top of each other to reach the final solution?
  - for example, refactorings of existing TestMacros or TestCase/TestSuite headers could be done before adding your special sauce to them. such as changing the current_test_case_did_fail function to set_test_case_result(Test::Failure), or adding test_result_to_string and such in TestMain
  - As it is, it is one 1800 line chunk that is hard to review
- C++ comments are preferred always. We should only have C-style comments (`/* */`) in the license header (for... aesthetic reasons? I actually don't know why we use that style there...)
- Full length name identifiers are preferred to shorthands in most cases. RandSource --> RandomSource, fn --> function etc
- The namespace of classes/functions should generally match the folder layout. In LibTest `namespace Test {}` in

LibTest: Add support for randomized tests #21191

Merged · ADKaster merged 13 commits into `SerenityOS:master` from `Janiczek:property-based-tests` on Oct 27, 2023

Conversation 112 · Commits 13 · Checks 13 · Files changed 16

Janiczek commented on Sep 22, 2023 • edited · Contributor

Add a way to run randomized tests (commonly called "property-based"): that is, tests that generate random data to run the test case with, and if they find a failure they shrink the input to a minimal failing example before reporting it to the user.

See README.md for more in-depth description.

Examples:

```
// Tests/LibCompress/TestBrid.cpp
// This test didn't find anything but shows off a very common property we can test with this.
RANDOMIZED_TEST_CASE(roundtrip)
{
    GEN(buffer, Gen::vector(240_kB,[]{return (u8)Gen::unsigned_int(256);}));
    auto const compressed = MUST(Compress::GzipCompressor::compress_all(buffer));
    auto const decompressed = MUST(Compress::GzipDecompressor::decompress_all(compressed));
    EXPECT(buffer == decompressed);
}
```

```
// Tests/LibIMAP/TestQuotedPrintable.cpp
// The randomized test below found a deviation from the spec!

TEST_CASE(section_6_7_3_white_space_regressions)
{
    // Found by the randomized test below.

    // Throws the encoded tab/space at the end of the string away
    DECODE_EQUAL(""_b*ex, ""sv);
    DECODE_EQUAL(" *ex,  "="sv);

    // Doesn't throw the encoded tab/space in the middle of the string away
    DECODE_EQUAL("_b*ex, "_b"sv);
    DECODE_EQUAL(" *ex,  "= *ex);
}

RANDOMIZED_TEST_CASE(section_6_7_3_white_space)
{
    /* https://datatracker.ietf.org/doc/html/rfc2049#section-6.7

       (3) (White Space) Octets with values of 9 and 32 MAY be
           represented as US-ASCII TAB (HT) and SPACE characters,
           respectively, but MUST NOT be so represented at the end
           of an encoded line.  Any TAB (HT) or SPACE characters
           on an encoded line MUST thus be followed on that line
           by a printable character.  In particular, an "=" at the
           end of an encoded line, indicating a soft line break
           (see rule #5) may follow one or more TAB (HT) or SPACE
           characters.  It follows that an octet with decimal
           value 9 or 32 appearing at the end of an encoded line
           must be represented according to Rule #1.  This rule is
           necessary because some MTAs (Message Transport Agents,
           programs which transport messages from one user to
           another, or perform a portion of such transfers) are
           known to pad lines of text with SPACEs, and others are
           known to remove "white space" characters from the end
           of a line.  Therefore, when decoding a Quoted-Printable
           body, any trailing white space on a line must be
           deleted, as it will necessarily have been added by
           intermediate transport agents.
    */

    GEN(prefix, literals_gen());
    auto prefix_sv = vector_to_string_view(prefix);

    // Throws the encoded tab at the end of the string away
    StringBuilder tab_at_end;
    tab_at_end.append(prefix_sv);
    tab_at_end.append('\t');
    DECODE_EQUAL(tab_at_end.string_view(), prefix_sv);

    // Throws the encoded space at the end of the string away
    StringBuilder space_at_end;
    space_at_end.append(prefix_sv);
    space_at_end.append(' ');
    DECODE_EQUAL(space_at_end.string_view(), prefix_sv);

    GEN(suffix, literals_gen());
    auto suffix_sv = vector_to_string_view(suffix);

    // Doesn't throw the encoded tab in the middle of the string away
    StringBuilder tab_in_middle;
    tab_in_middle.append(prefix_sv);
    tab_in_middle.append('\t');
    tab_in_middle.append(suffix_sv);
    StringView tab_in_middle_sv = tab_in_middle.string_view();
    DECODE_EQUAL(tab_in_middle_sv, tab_in_middle_sv);

    // Doesn't throw the encoded space in the middle of the string away
    StringBuilder space_in_middle;
    space_in_middle.append(prefix_sv);
    space_in_middle.append(' ');
    space_in_middle.append(suffix_sv);
    StringView space_in_middle_sv = space_in_middle.string_view();
    DECODE_EQUAL(space_in_middle_sv, space_in_middle_sv);
}
```

(I'll make another PR with usages of `RANDOMIZED_TEST_CASE` that will build on this one; I appreciate this PR is already big as it is!)

> **Note**
>
> This being my first SerenityOS contribution, I'll be glad for any suggestions (code style, C++ tricks, namespace organization)!

github-actions bot added the `pr-needs-review` label on Sep 22, 2023

BuggieBot commented on Sep 22, 2023 · Member

Hello!

One or more of the commit messages in this PR do not match the SerenityOS code submission policy, please check the `Lint_commits` CI job for more details on which commits were flagged and why.
Please do not close this PR and open another, instead modify your commit message(s) with git commit --amend and force push those changes to update this PR.

Janiczek force-pushed the `property-based-tests` branch from `3b76be5` to `11a3b3c` 5 months ago · Compare

ADKaster commented on Sep 23, 2023 · Member

Before taking a real look at this, a few general comments from a scroll through:

- Make sure to fixup-squash any updates, or PR review comment changes.
- As written, this is one huge commit.
  - Can you split it into "atomic commits", that each change one thing, and build on top of each other to reach the final solution?
  - for example, refactorings of existing TestMacros or TestCase/TestSuite headers could be done before adding your special sauce to them. such as changing the current_test_case_did_fail function to set_test_case_result(Test::Failure), or adding test_result_to_string and such in TestMain
  - As it is, it is one 1800 line chunk that is hard to review
- C++ comments are preferred always. We should only have C-style comments (`/* */`) in the license header (for... aesthetic reasons? I actually don't know why we use that style there...)
- Full length name identifiers are preferred to shorthands in most cases. RandSource --> RandomSource, fn --> function etc
- The namespace of classes/functions should generally match the folder layout. In LibTest/ `namespace Test {}` in

LibTest/Randomized/ `namespace Test::Randomized` etc.

- Template parameter names should generally follow the same naming scheme as normal classes, with the obvious exception of T, U, etc (single char) (looking at you, SET_FN, and FN)
- static inline functions in headers are no bueno. It's a nice trap to duplicate the function into every TU independently.
- I wonder if your ASSUME/REJECT etc error cases could be modeled with `AK::IterationError`, `GenerationError`, our common result/std::expected-like type?
  - At least one of them, for `one_of` could just be a `requires` clause on the end of the template function declaration `) requires ( sizeof...(Ts) > 0; )` or similar. We are in C++-20 after all.
- Some of your algorithms bear a striking resemblance to ones in AK. `one_of` comes to mind again, @tdm5180 added `any_of` in AK/AnyOf a while back, but we've been missing that drive to add generic algorithms lately...

.. ok maybe these aren't generic, but those things are all a bit distracting from the cool feature you've added to let us validate properties of our libraries without relying on oss-fuzz :)

Janiczek force-pushed the `property-based-tests` branch 7 times, most recently from `ab4f4fb` to `59d1609` 5 months ago · Compare

Janiczek mentioned this pull request on Oct 11, 2023

AK: Fix one-off error in BitmapView::find_first and find_one_anywhere #21409 Merged

timschumi self-requested a review 4 months ago

timschumi requested changes on Oct 11, 2023 · View reviewed changes

timschumi left a comment · Member

First round of feedback, mostly style-related.

- `Userland/Libraries/LibTest/TestResult.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/Chunk.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/RandomRun.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/RandomRun.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/RandomRun.h` Outdated · Show resolved

11 hidden conversations — Load more...

- `Userland/Libraries/LibTest/Randomized/ShrinkCmd.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/ShrinkCmd.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/Shrink.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Macros.h` Outdated · Show resolved
- `Tests/LibTest/TestGenerator.cpp` Outdated · Show resolved

Janiczek force-pushed the `property-based-tests` branch from `99d1609` to `4f672e6` 4 months ago · Compare

Janiczek requested a review from **timschumi** 4 months ago

Janiczek force-pushed the `property-based-tests` branch 3 times, most recently from `ec41270` to `1f6a20c` 4 months ago · Compare

ADKaster requested changes on Oct 18, 2023 · View reviewed changes

ADKaster left a comment · Member

All right, here's my nitpicks. I think most of them should be trivial, but there are a few thinker questions hidden in the noise :]

- `Userland/Libraries/LibTest/Macros.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/TestSuite.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/Chunk.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/RandomRun.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/RandomRun.h` Outdated · Show resolved

6 hidden conversations — Load more...

- `Userland/Libraries/LibTest/Randomized/ShrinkCommand.h` · Show resolved
- `Userland/Libraries/LibTest/Randomized/ShrinkCommand.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/ShrinkCommand.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/Randomized/Shrink.h` Outdated · Show resolved
- `Userland/Libraries/LibTest/TestCase.h` Outdated · Show resolved

Janiczek force-pushed the `property-based-tests` branch from `1f6a20c` to `92d0a37` 4 months ago · Compare

Janiczek requested a review from **ADKaster** 4 months ago

Janiczek force-pushed the `property-based-tests` branch from `92d0a37` to `2e0cc93` 4 months ago · Compare

timschumi requested changes on Oct 20, 2023 · View reviewed changes

timschumi left a comment · Member

If any of the comments here have already been resolved in advance feel free to disregard them, the first few I started writing a some days ago.

LibTest: Add support for randomized tests #21191

Merged · ADKaster merged 13 commits into SerenityOS:master from Janiczek:property-based-tests on Oct 27, 2023

Conversation 112 · Commits 13 · Checks 13 · Files changed 16

Janiczek commented on Sep 22, 2023 · edited · Contributor

Add a way to run randomized tests (commonly called "property-based"): that is, tests that generate random data to run the test case with, and if they find a failure they shrink the input to a minimal failing example before reporting it to the user.

See README.md for more in-depth description.

Examples:

```
// Tests/LibCompress/TestBrid.cpp
// This test didn't find anything but shows off a very common property we can test with this.
RANDOMIZED_TEST_CASE(roundtrip)
{
    GEN(buffer, Gen::vector(2048, [](){return (u8)Gen::unsigned_int(256);}));
    auto const compressed = MUST(Compress::BridCompressor::compress_all(buffer));
    auto const decompressed = MUST(Compress::BridDecompressor::decompress_all(compressed));
    EXPECT(buffer == decompressed);
}
```

```
// Tests/LibIMAP/TestQuotedPrintable.cpp
// The randomized test below found a deviation from the spec!

TEST_CASE(section_6_7_3_white_space_regressions)
{
    // Found by the randomized test below.

    // Throws the encoded tab/space at the end of the string away
    DECODE_EQUAL(""_b*ex,  ""sv);
    DECODE_EQUAL(" *ex,  ""sv);

    // Doesn't throw the encoded tab/space in the middle of the string away
    DECODE_EQUAL("\t\t"ex, "\t\t"sv);
    DECODE_EQUAL(" *ex,  " *ex);
}

RANDOMIZED_TEST_CASE(section_6_7_3_white_space)
{
    /* https://datatracker.ietf.org/doc/html/rfc2045#section-6.7

       (3) (White Space) Octets with values of 9 and 32 MAY be
           represented as US-ASCII TAB (HT) and SPACE characters,
           respectively, but MUST NOT be so represented at the end
           of an encoded line.  Any TAB (HT) or SPACE characters
           on an encoded line MUST thus be followed on that line
           by a printable character.  In particular, an "=" at the
           end of an encoded line, indicating a soft line break
           (see rule #5) may follow one or more TAB (HT) or SPACE
           characters.  It follows that an octet with decimal
           value 9 or 32 appearing at the end of an encoded line
           must be represented according to Rule #1.  This rule is
           necessary because some MTAs (Message Transport Agents,
           programs which transport messages from one user to
           another, or perform a portion of such transfers) are
           known to pad lines of text with SPACEs, and others are
           known to remove "white space" characters from the end
           of a line.  Therefore, when decoding a Quoted-Printable
           body, any trailing white space on a line must be
           deleted, as it will necessarily have been added by
           intermediate transport agents.
    */

    GEN(prefix, literals_gen());
    auto prefix_sv = vector_to_string_view(prefix);

    // Throws the encoded tab at the end of the string away
    StringBuilder tab_at_end;
    tab_at_end.append(prefix_sv);
    tab_at_end.append('\t');
    DECODE_EQUAL(tab_at_end.string_view(), prefix_sv);

    // Throws the encoded space at the end of the string away
    StringBuilder space_at_end;
    space_at_end.append(prefix_sv);
    space_at_end.append(' ');
    DECODE_EQUAL(space_at_end.string_view(), prefix_sv);

    GEN(suffix, literals_gen());
    auto suffix_sv = vector_to_string_view(suffix);

    // Doesn't throw the encoded tab in the middle of the string away
    StringBuilder tab_in_middle;
    tab_in_middle.append(prefix_sv);
    tab_in_middle.append('\t');
    tab_in_middle.append(suffix_sv);
    StringView tab_in_middle_sv = tab_in_middle.string_view();
    DECODE_EQUAL(tab_in_middle_sv, tab_in_middle_sv);

    // Doesn't throw the encoded space in the middle of the string away
    StringBuilder space_in_middle;
    space_in_middle.append(prefix_sv);
    space_in_middle.append(' ');
    space_in_middle.append(suffix_sv);
    StringView space_in_middle_sv = space_in_middle.string_view();
    DECODE_EQUAL(space_in_middle_sv, space_in_middle_sv);
}
```

(I'll make another PR with usages of `RANDOMIZED_TEST_CASE` that will build on this one; I appreciate this PR is already big as it is!)

> **Note**
>
> This being my first SerenityOS contribution, I'll be glad for any suggestions (code style, C++ tricks, namespace organization)!

github-actions (bot) added the pr-needs-review label on Sep 22, 2023

BuggieBot commented on Sep 22, 2023 · Member

Hello!

One or more of the commit messages in this PR do not match the SerenityOS code submission policy, please check the `Lint Commits` CI job for more details on which commits were flagged and why.
Please do not close this PR and open another, instead modify your commit message(s) with git commit --amend and force push those changes to update this PR.

Janiczek force-pushed the property-based-tests branch from 3b76be5 to 11a3b3c 5 months ago · Compare

ADKaster commented on Sep 23, 2023 · Member

Before taking a real look at this, a few general comments from a scroll through:

- Make sure to fixup-squash any updates, or PR review comment changes.
- As written, this is one huge commit.
  - Can you split it into "atomic commits", that each change one thing, and build on top of each other to reach the final solution?
  - for example, refactorings of existing TestMacros or TestCase/TestSuite headers could be done before adding your special sauce to them. such as changing the current_test_case_did_fail function to set_test_case_result(Test::Failure), or adding test_result_to_string and such in TestMain
  - As it is, it is one 1800 line chunk that is hard to review
- C++ comments are preferred always. We should only have C-style comments (`/* */`) in the license header (for... aesthetic reasons? I actually don't know why we use that style there...)
- Full length name identifiers are preferred to shorthands in most cases. RandSource --> RandomSource, fn --> function etc
- The namespace of classes/functions should generally match the folder layout. In LibTest/ `namespace Test {}` in

LibTest(Randomized)? `namespace Test::Randomized` etc.
- Template parameter names should generally follow the same naming scheme as normal classes, with the obvious exception of T, U, etc (single char) (looking at you, SET_FN, and FN)
- static inline functions in headers are no bueno. It's a nice trap to duplicate the function into every TU independently.
- I wonder if your ASSUME/REJECT etc error cases could be modeled with `AK::ErrorOr<T, GenerationError>`, our common result/std::expected-like type?
  - At least one of them, for `one_of` could just be a `requires` clause on the end of the template function declaration `{ requires ( sizeof...(Ts) > 0 ) }` or similar. We are in C++-20 after all.
- Some of your algorithms bear a striking resemblance to ones in AK. `one_of` comes to mind again. @tdm5180 added `any_of` in AK/AnyOf a while back, but we've been missing that drive to add generic algorithms lately...

...ok maybe these aren't generic. but those things are all a bit distracting from the cool feature you've added to let us validate properties of our libraries without relying on oss-fuzz :)

Janiczek force-pushed the property-based-tests branch 7 times, most recently from ab4f4b to 59d1e09 · Compare · 5 months ago

Janiczek mentioned this pull request on Oct 11, 2023

AK: Fix one-off error in BitmapView::find_first and find_one_anywhere #21409 · Merged

timschumi self-requested a review 4 months ago

timschumi requested changes on Oct 11, 2023 · View reviewed changes

timschumi left a comment · Member

First round of feedback, mostly style-related.

Userland/Libraries/LibTest/TestResult.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/Chunk.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/RandomRun.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/RandomRun.cpp · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/RandomRun.h · Outdated · Show resolved

11 hidden conversations
Load more...

Userland/Libraries/LibTest/Randomized/ShrinkCmd.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/ShrinkCmd.cpp · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/Shrink.h · Outdated · Show resolved
Userland/Libraries/LibTest/Macros.h · Outdated · Show resolved
Tests/LibTest/TestGenerator.cpp · Outdated · Show resolved

Janiczek force-pushed the property-based-tests branch from 59d1e09 to 4f672e6 4 months ago · Compare

Janiczek requested a review from **timschumi** 4 months ago

Janiczek force-pushed the property-based-tests branch 3 times, most recently from ec41270 to 1f6a20c · Compare · 4 months ago

ADKaster requested changes on Oct 18, 2023 · View reviewed changes

ADKaster left a comment · Member

All right, here's my nitpicks. I think most of them should be trivial, but there are a few thinker questions hidden in the noise :]

Userland/Libraries/LibTest/Macros.h · Outdated · Show resolved
Userland/Libraries/LibTest/TestSuite.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/Chunk.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/RandomRun.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/RandomRun.h · Outdated · Show resolved

6 hidden conversations
Load more...

Userland/Libraries/LibTest/Randomized/ShrinkCommand.h · Show resolved
Userland/Libraries/LibTest/Randomized/ShrinkCommand.cpp · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/ShrinkCommand.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/Shrink.h · Outdated · Show resolved
Userland/Libraries/LibTest/TestCase.h · Outdated · Show resolved

Janiczek force-pushed the property-based-tests branch from 1f6a20c to 92dba37 4 months ago · Compare

Janiczek requested a review from **ADKaster** 4 months ago

Janiczek force-pushed the property-based-tests branch from 92dba37 to 2e0cc93 4 months ago · Compare

timschumi requested changes on Oct 20, 2023 · View reviewed changes

timschumi left a comment · Member

If any of the comments here have already been resolved in advance feel free to disregard them, the first few I started writing a some days ago.

Userland/Libraries/LibTest/Macros.h · Show resolved
Userland/Libraries/LibTest/TestResult.h · Outdated · Show resolved
Userland/Libraries/LibTest/TestSuite.cpp · Show resolved
Userland/Libraries/LibTest/Randomized/Chunk.h · Outdated · Show resolved
Userland/Libraries/LibTest/Randomized/RandomRun.h · Outdated · Show resolved

9 hidden conversations
Load more...

Userland/Libraries/LibTest/Randomized/Shrink.h · Outdated · Show resolved
Userland/Libraries/LibTest/TestCase.h · Show resolved

Comment on lines +28 to +29

```
+ #ifndef MAX_GENERATED_VALUES_PER_TEST
+ #    define MAX_GENERATED_VALUES_PER_TEST 100
+ #endif
+
+ #ifndef MAX_GEN_ATTEMPTS_PER_VALUE
+ #    define MAX_GEN_ATTEMPTS_PER_VALUE 10
+ #endif
```

timschumi on Oct 20, 2023 · Member

These look like they simply might want to be local `static constexpr size_t` or something like that.

Janiczek on Oct 20, 2023 · Contributor · Author

How do we override them later though? Do we want to make the test runner accept a cmdline flag?

timschumi on Oct 24, 2023 · edited · Member

If they are supposed to be changeable at run time, then `constexpr` really isn't the right choice either.

But in any case, needing to change these at build time by manually editing build flags to pass definitions is weird, if they are meant to be changed often. If they aren't meant to be changed often, then having them as a `constexpr` value that has to be edited in the source code is probably fine, certainly for a first pass.

Janiczek on Oct 24, 2023 · Contributor · Author

It's useful in practice to have easy access to changing `MAX_GENERATED_VALUES_PER_TEST`. Sometimes you want to run eg. 10k tries instead of 100, just to get extra confidence.

In comparison, I've never needed to change `MAX_GEN_ATTEMPTS_PER_VALUE` and that one perhaps could be a constexpr.

I'll make the first one a runtime flag (similarly to how `--benchmark-repetitions N` is done) and the second one a constexpr.

Reply...

Resolve conversation

Userland/Libraries/LibTest/TestCase.h · Outdated · Show resolved
Userland/Libraries/LibTest/TestCase.h · Show resolved
Userland/Libraries/LibTest/Randomized/README.md

Comment on lines +75 to +158

```
+ ## Code organization
+
+ - `TestResult.h`
+   - Defines an enum class TestResult.
+     This expands the typical "passed / failed": we also need to care about
+     a generator rejecting a RandomRun (eg. when the user calls the ASSUME(...)
+     macro with a predicate that can't be satisfied).
+
+ - `Generator.h`
+   - Contains generators: fns of shape f(), eg. `u32 Gen::unsigned_int(u32 max)`
+     - These implicitly depend on a RandomnessSource held by the singleton
+       TestSuite.
+   - These can be called directly, but the top-level use by the user should always
+     happen via the GEN(...) macro which makes sure the generated value gets
+     logged to the user in case of a failure.
+   - Example:
+     `Gen::vector(1, x, []() { return Gen::unsigned_int(5); })`
+     generates vectors of length between 1 and x, of unsigned ints in range 0..5.
+     Eg. `{2,5,1}`, `{0}`, `{3,5,5,2}`.
+
+ - `RandomnessSource.h`
+   - A source of random bits.
+   - There are two variants of `RandomnessSource`:
+     - Live: gives `AK/Random.h` u32 values and remembers them into a `RandomRun`
+     - Recorded: gives (replays) u32 values from a static `RandomRun`
+
+ - `RandomRun.h`
+   - A finite sequence of random bits (in practice, `u32`s).
+   - Example: `{2,5,6,11,0,0,0,1}`
+
+ - `ShrinkCommand.h`
+   - A high-level recipe for how to try and minimize a given `RandomRun`.
+   - For example, "zero this contiguous chunk of it" or "minimize the number on
+     this index using binary search".
+   - These later get interpreted by the FMT runner on a specific `RandomRun`.
+
+ - `Chunk.h`
+   - A description of a contiguous `RandomRun`'slice.
+   - Example: `Chunk{size = 4, index = 1}` : `[_,X,X,X,_,_]`
+
+ - `Shrink.h`
+   - Algorithms for interpreting `ShrinkCommand`s and the main shrinking loop
+
+ - `TestCase.h`
+   - The `TestCase::run(mut(...)` function contains the main testing loop
```

timschumi on Oct 20, 2023 · Member

While we appreciate the effort, to me it looks like the comments within the actual files will be more than sufficient for explaining the structure - duplicating the information here will probably just open up opportunities for the documentation to be outdated, and they are probably closer to internals anyways.

ADKaster on Oct 20, 2023 · Member

If we really want the file, it could live in Documentation/.

Janiczek on Oct 20, 2023 · Contributor · Author

I felt like a summary "all in one place" would be helpful to get an overall picture, but I can remove it if you don't feel like it helped you much.

I was also contemplating another file with more hands-on examples of using this library, explaining the macros and so on. Examples of writing generators, writing the tests, etc... WDYT?

timschumi on Oct 24, 2023 · Member

Personally, I don't mind the "Example", "Property based testing?" and "Implementation" parts, but as Andrew

LibTest: Add support for randomized tests #21191

Merged — ADKaster merged 13 commits into SerenityOS:master from Janiczek:property-based-tests on Oct 27, 2023

Conversation · Commits · Checks · Files changed

Janiczek commented on Sep 22, 2023 • edited

Add a way to run randomized tests (commonly called "property-based"): that is, tests that generate random data to run the test case with, and if they find a failure they shrink the input to a minimal failing example before reporting it to the user.

See README.md for more in-depth description.

Examples:

```
// Tests/LibCompress/TestGzip.cpp
// This test didn't find anything but shows off a very common property we can test with this.
RANDOMIZED_TEST_CASE(roundtrip)
{
    GEN(buffer, Gen::vector(2048, [](){ return (u8)Gen::unsigned_int(255); }));
    auto const compressed = MUST(Compress::GzipCompressor::compress_all(buffer));
    auto const decompressed = MUST(Compress::GzipDecompressor::decompress_all(compressed));
    EXPECT(buffer == decompressed);
}
```

```
// Tests/LibIMAP/TestQuotedPrintable.cpp
// The randomized test below found a deviation from the spec!

TEST_CASE(section_6_7_3_white_space_regressions)
{
    // Found by the randomized test below.

    // Throws the encoded tab/space at the end of the string away
    ...
}

RANDOMIZED_TEST_CASE(section_6_7_3_white_space)
{
    ...
}
```

(I'll make another PR with usages of RANDOMIZED_TEST_CASE that will build on this one; I appreciate this PR is already big as is!)

Note

This being my first SerenityOS contribution, I'll be glad for any suggestions (code style, C++ tricks, namespace organization)!

github-actions (bot) added the 👀 pr-needs-review label on Sep 22, 2023

BuggieBot commented on Sep 22, 2023

Hello!

One or more of the commit messages in this PR do not match the SerenityOS code submission policy, please check the lint_commits CI job for more details on which commits were flagged and why.
Please do not close this PR and open another, instead modify your commit message(s) with git commit --amend and force push those changes to update this PR.

Janiczek force-pushed the property-based-tests branch 5 months ago

ADKaster commented on Sep 23, 2023

Before taking a real look at this, a few general comments from a scroll through:

- Make sure to fixup-squash any updates, or PR review comment changes.
- As written, this is one huge commit.
  - Can you split it into "atomic commits", that each change one thing, and build on top of each other to reach the final solution?
  - for example, refactorings of existing TestMacros or TestCase/TestSuite headers could be done before adding your special sauce to them.
  - As it is, it is one 1800 line chunk that is hard to review
- C++ comments are preferred always. We should only have C-style comments in the license header.
- Full length name identifiers are preferred to shorthands in most cases. RandSource --> RandomSource, fn --> function etc
- The namespace of classes/functions should generally match the folder layout.

...

- static inline functions in headers are no bueno.
- Some of your algorithms bear a striking resemblance to ones in AK.

...ok maybe these aren't generic, but those things are all a bit distracting from the cool feature you've added to let us validate properties of our libraries without relying on oss-fuzz :)

Janiczek force-pushed the property-based-tests branch 7 times

Janiczek mentioned this pull request on Oct 11, 2023
AK: Fix one-off error in BitmapView::find_first and find_one_anywhere #21409 — Merged

timschumi self-requested a review 4 months ago

timschumi requested changes on Oct 11, 2023

First round of feedback, mostly style-related.

Janiczek requested a review from timschumi 4 months ago

ADKaster requested changes on Oct 18, 2023

All right, here's my nitpicks. I think most of them should be trivial, but there are a few thinker questions hidden in the noise :)

Janiczek requested a review from ADKaster 4 months ago

timschumi requested changes on Oct 20, 2023

If any of the comments here have already been resolved in advance feel free to disregard them, the first few I started writing a some days ago.

Userland/Libraries/LibTest/Randomized/README.md

timschumi on Oct 20, 2023

These look like they simply might want to be local static constexpr size_t or something like that.

Janiczek on Oct 20, 2023

How do we override them later though? Do we want to make the test runner accept a cmdline flag?

timschumi on Oct 24, 2023

If they are supposed to be changeable at run time, then constexpr really isn't the right choice either.

Janiczek on Oct 24, 2023

I'll make the first one a runtime flag and the second one a constexpr.

Resolve conversation

timschumi on Oct 20, 2023

While we appreciate the effort, to me it looks like the comments within the actual files will be more than sufficient for explaining the structure - duplicating the information here will probably just open up opportunities for the documentation to be outdated, and they are probably closer to internals anyways.

ADKaster on Oct 20, 2023

If we really want the file, it could live in Documentation/.

Janiczek on Oct 20, 2023

I felt like a summary "all in one place" would be helpful to get an overall picture, but I can remove it if you don't feel like it helped you much.

ADKaster reviewed on Oct 20, 2023

Janiczek force-pushed the property-based-tests branch 3 times

Janiczek requested review from ADKaster and timschumi 4 months ago

timschumi reviewed on Oct 24, 2023

ADKaster commented on Oct 27, 2023

@Janiczek I'm going to push two fixes for this and then merge so we can iterate in tree. thanks for being so patient on the review!

Janiczek added 13 commits 4 months ago

- LibTest: Add the Chunk abstraction
- LibTest: Add the RandomRun abstraction
- LibTest: Add the RandomnessSource abstraction
- LibTest: Add a library of generators
- LibTest: Add the REJECT and ASSUME macros
- LibTest: Add the GEN macro
- LibTest: Add the ShrinkCommand abstraction
- LibTest: Add functions for shrinking a given RandomRun
- LibTest: Add the RANDOMIZED_TEST_CASE macro and its main loop
- LibTest: Add a suite of tests for the generators
- LibTest: Add a README documenting the high-level randomized approach

ADKaster force-pushed the property-based-tests branch 4 months ago

ADKaster approved these changes on Oct 27, 2023

ADKaster merged commit into SerenityOS:master on Oct 27, 2023

github-actions (bot) removed the 👀 pr-needs-review label on Oct 27, 2023

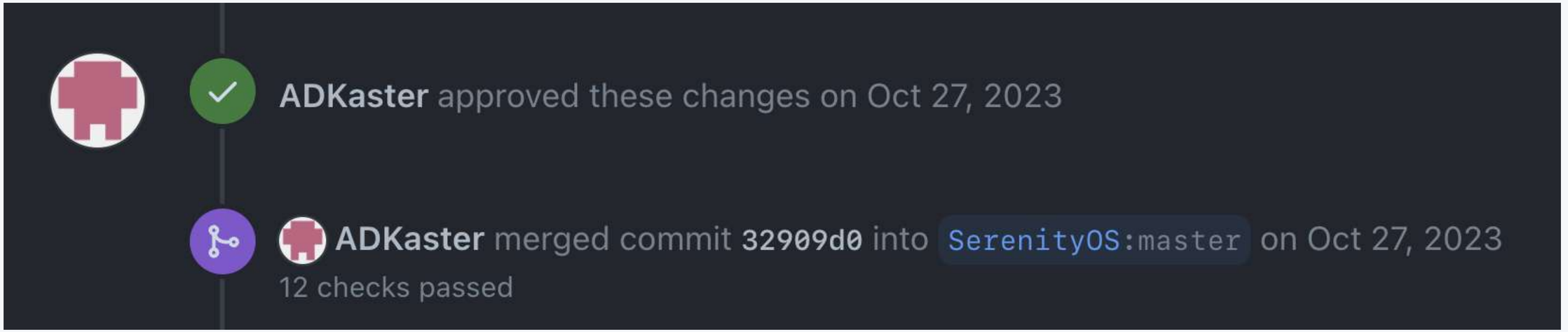
Janiczek deleted the property-based-tests branch 4 months ago

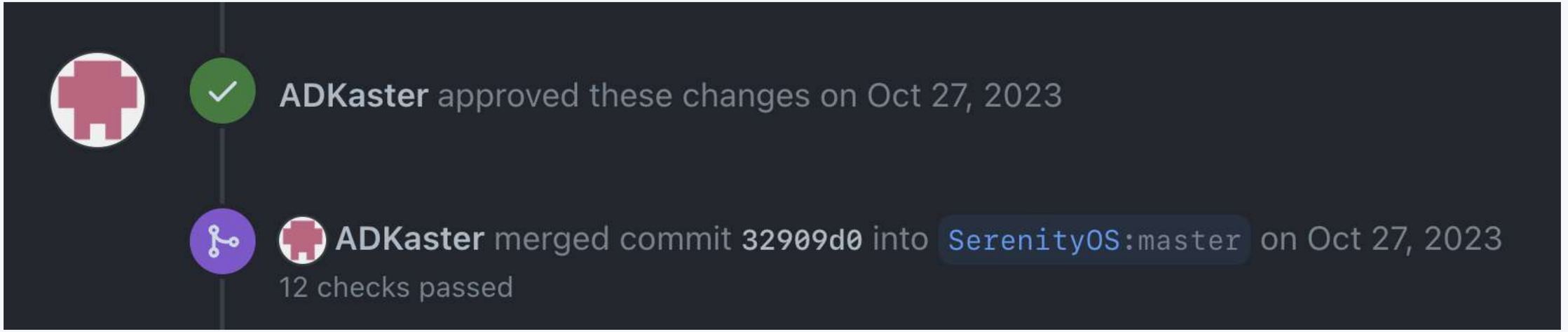
Add a comment

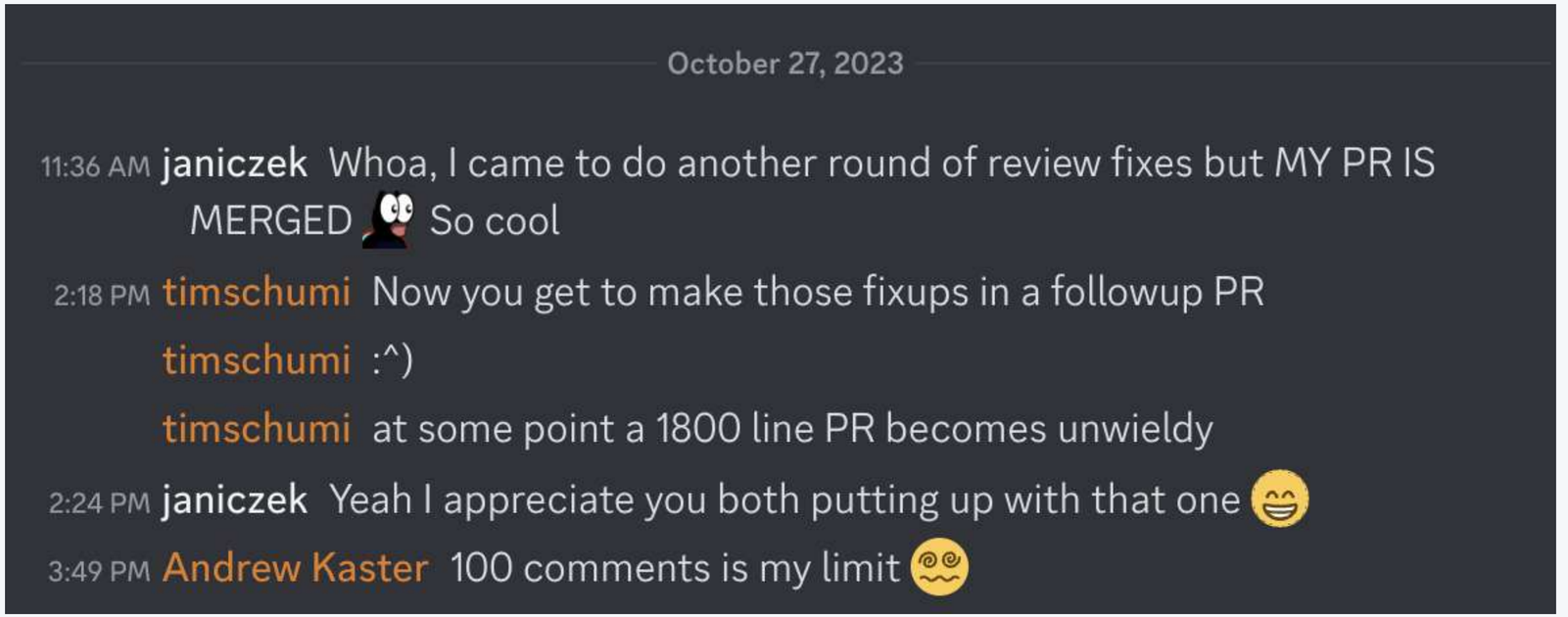
Write · Preview

Markdown is supported · Paste, drop, or click to add files

Comment


ADKaster approved these changes on Oct 27, 2023
ADKaster merged commit 32909d0 into SerenityOS:master on Oct 27, 2023
12 checks passed


ADKaster approved these changes on Oct 27, 2023
ADKaster merged commit 32909d0 into SerenityOS:master on Oct 27, 2023
12 checks passed


October 27, 2023
11:36 AM janiczek Whoa, I came to do another round of review fixes but MY PR IS MERGED So cool
2:18 PM timschumi Now you get to make those fixups in a followup PR
timschumi :^)
timschumi at some point a 1800 line PR becomes unwieldy
2:24 PM janiczek Yeah I appreciate you both putting up with that one
3:49 PM Andrew Kaster 100 comments is my limit

Examples:

```
// Tests/LibCompress/TestGzip.cpp
// This test didn't find anything but shows off a very common property we can test with this.
RANDOMIZED_TEST_CASE(roundtrip)
{
    GEN(buffer, Gen::vector(2048,[](){return (u8)Gen::unsigned_int(255);}));
    auto const compressed = MUST(Compress::GzipCompressor::compress_all(buffer));
    auto const decompressed = MUST(Compress::GzipDecompressor::decompress_all(compressed));
    EXPECT(buffer == decompressed);
}
```

Network Working Group
Request for Comments: 2045
Obsoletes: 1521, 1522, 1590
Category: Standards Track

N. Freed
Innosoft
N. Borenstein
First Virtual
November 1996

Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

STD 11, RFC 822, defines a message representation protocol specifying considerable detail about US-ASCII message headers, and leaves the message content, or message body, as flat US-ASCII text. This set of documents, collectively called the Multipurpose Internet Mail Extensions, or MIME, redefines the format of messages to allow for

(1) textual message bodies in character sets other than US-ASCII,

(2) an extensible set of different formats for non-textual message bodies,

(3) multi-part message bodies, and

(4) textual header information in character sets other than US-ASCII.

These documents are based on earlier work documented in RFC 934, STD

RFC 2045 - Multipurpose Inte

datatracker.ietf.org/doc/html/rfc2045#section-6.7

Network Working Group
Request for Comments: 2045
Obsoletes: 1521, 1522, 1590
Category: Standards Track

N. Freed
Innosoft
N. Borenstein
First Virtual
November 1996

**Multipurpose Internet Mail Extensions
(MIME) Part One:
Format of Internet Message Bodies**

AUDIENCE PARTICIPATION

RFC 2045, § 6.7 (3)

and status of this protocol. Distribution of this memo is unlimited.

(1) textual message bodies in character sets other than US-ASCII,

(2) an extensible set of different formats for non-textual message bodies,

(3) multi-part message bodies, and

(4) textual header information in character sets other than US-ASCII.

These documents are based on earlier work documented in RFC 934, STD

Datatracker

RFC 2045

Draft Standard

Info | Contents | Prefs

Document type

RFC Draft Standard

November 1996

WG)

Side-by-side | Inline

Authors

Ned Freed, Dr. Nathaniel S. Borenstein

Email authors

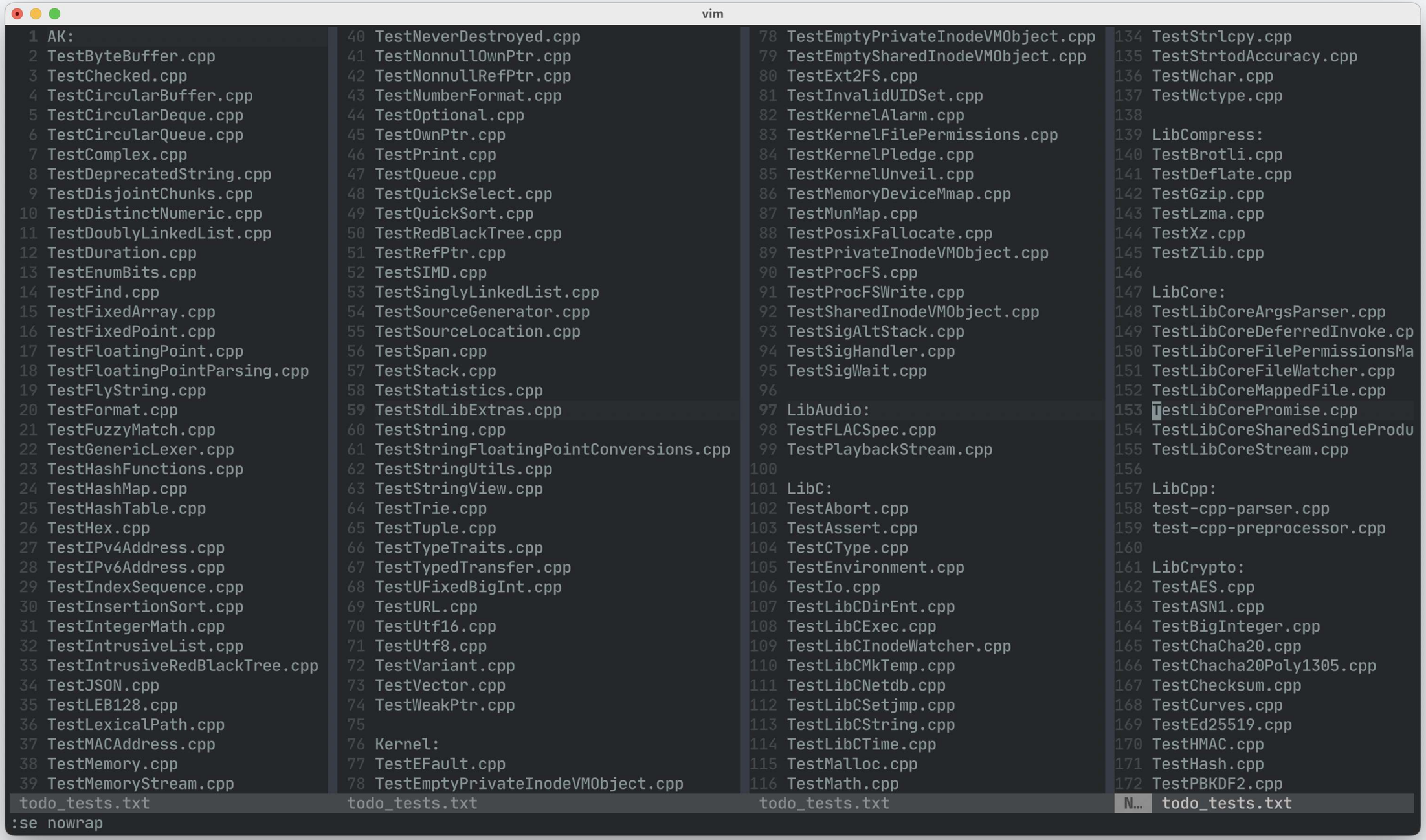
RFC stream

I E T F

```
GEN(prefix, literals_gen());
auto prefix_sv = vector_to_string_view(prefix);

// Throws the encoded tab at the end of the string away
StringBuilder tab_at_end;
tab_at_end.append(prefix_sv);
tab_at_end.append(9);
DECODE_EQUAL(tab_at_end.string_view(), prefix_sv);

// Throws the encoded space at the end of the string away
StringBuilder space_at_end;
space_at_end.append(prefix_sv);
space_at_end.append(32);
DECODE_EQUAL(space_at_end.string_view(), prefix_sv);
```

```
    GEN(prefix, literals_gen());
    auto prefix_sv = vector_to_string_view(prefix);

    // Throws the encoded tab at the end of the string away
```

```
// Tests/LibIMAP/TestQuotedPrintable.cpp
// The randomized test below found a deviation from the spec!

TEST_CASE(section_6_7_3_white_space_regressions)
{
    // Found by the randomized test below.

    // Throws the encoded tab/space at the end of the string away
    DECODE_EQUAL("!\t"sv,  "!"sv);
    DECODE_EQUAL("! "sv,   "!"sv);

    // Doesn't throw the encoded tab/space in the middle of the string away
    DECODE_EQUAL("!\t!"sv, "!\t!"sv);
    DECODE_EQUAL("! !"sv,  "! !"sv);
}
```


vim
1 AK:
2 TestByteBuffer.cpp
3 TestChecked.cpp
4 TestCircularBuffer.cpp
5 TestCircularDeque.cpp
6 TestCircularQueue.cpp
7 TestComplex.cpp
8 TestDeprecatedString.cpp
9 TestDisjointChunks.cpp
10 TestDistinctNumeric.cpp
11 TestDoublyLinkedList.cpp
12 TestDuration.cpp
13 TestEnumBits.cpp
14 TestFind.cpp
15 TestFixedArray.cpp
16 TestFixedPoint.cpp
17 TestFloatingPoint.cpp
18 TestFloatingPointParsing.cpp
19 TestFlyString.cpp
20 TestFormat.cpp
21 TestFuzzyMatch.cpp
22 TestGenericLexer.cpp
23 TestHashFunctions.cpp
24 TestHashMap.cpp
25 TestHashTable.cpp
26 TestHex.cpp
27 TestIPv4Address.cpp
28 TestIPv6Address.cpp
29 TestIndexSequence.cpp
30 TestInsertionSort.cpp
31 TestIntegerMath.cpp
32 TestIntrusiveList.cpp
33 TestIntrusiveRedBlackTree.cpp
34 TestJSON.cpp
35 TestLEB128.cpp
36 TestLexicalPath.cpp
37 TestMACAddress.cpp
38 TestMemory.cpp
39 TestMemoryStream.cpp
todo_tests.txt
40 TestNeverDestroyed.cpp
41 TestNonnullOwnPtr.cpp
42 TestNonnullRefPtr.cpp
43 TestNumberFormat.cpp
44 TestOptional.cpp
45 TestOwnPtr.cpp
46 TestPrint.cpp
47 TestQueue.cpp
48 TestQuickSelect.cpp
49 TestQuickSort.cpp
50 TestRedBlackTree.cpp
51 TestRefPtr.cpp
52 TestSIMD.cpp
53 TestSinglyLinkedList.cpp
54 TestSourceGenerator.cpp
55 TestSourceLocation.cpp
56 TestSpan.cpp
57 TestStack.cpp
58 TestStatistics.cpp
59 TestStdLibExtras.cpp
60 TestString.cpp
61 TestStringFloatingPointConversions.cpp
62 TestStringUtils.cpp
63 TestStringView.cpp
64 TestTrie.cpp
65 TestTuple.cpp
66 TestTypeTraits.cpp
67 TestTypedTransfer.cpp
68 TestUFixedBigInt.cpp
69 TestURL.cpp
70 TestUtf16.cpp
71 TestUtf8.cpp
72 TestVariant.cpp
73 TestVector.cpp
74 TestWeakPtr.cpp
75
76 Kernel:
77 TestEFault.cpp
78 TestEmptyPrivateInodeVMObject.cpp
todo_tests.txt
78 TestEmptyPrivateInodeVMObject.cpp
79 TestEmptySharedInodeVMObject.cpp
80 TestExt2FS.cpp
81 TestInvalidUIDSet.cpp
82 TestKernelAlarm.cpp
83 TestKernelFilePermissions.cpp
84 TestKernelPledge.cpp
85 TestKernelUnveil.cpp
86 TestMemoryDeviceMmap.cpp
87 TestMunMap.cpp
88 TestPosixFallocate.cpp
89 TestPrivateInodeVMObject.cpp
90 TestProcFS.cpp
91 TestProcFSWrite.cpp
92 TestSharedInodeVMObject.cpp
93 TestSigAltStack.cpp
94 TestSigHandler.cpp
95 TestSigWait.cpp
96
97 LibAudio:
98 TestFLACSpec.cpp
99 TestPlaybackStream.cpp
100
101 LibC:
102 TestAbort.cpp
103 TestAssert.cpp
104 TestCType.cpp
105 TestEnvironment.cpp
106 TestIo.cpp
107 TestLibCDirEnt.cpp
108 TestLibCExec.cpp
109 TestLibCInodeWatcher.cpp
110 TestLibCMkTemp.cpp
111 TestLibCNetdb.cpp
112 TestLibCSetjmp.cpp
113 TestLibCString.cpp
114 TestLibCTime.cpp
115 TestMalloc.cpp
116 TestMath.cpp
todo_tests.txt
134 TestStrlcpy.cpp
135 TestStrtodAccuracy.cpp
136 TestWchar.cpp
137 TestWctype.cpp
138
139 LibCompress:
140 TestBrotli.cpp
141 TestDeflate.cpp
142 TestGzip.cpp
143 TestLzma.cpp
144 TestXz.cpp
145 TestZlib.cpp
146
147 LibCore:
148 TestLibCoreArgsParser.cpp
149 TestLibCoreDeferredInvoke.cp
150 TestLibCoreFilePermissionsMa
151 TestLibCoreFileWatcher.cpp
152 TestLibCoreMappedFile.cpp
153 TestLibCorePromise.cpp
154 TestLibCoreSharedSingleProdu
155 TestLibCoreStream.cpp
156
157 LibCpp:
158 test-cpp-parser.cpp
159 test-cpp-preprocessor.cpp
160
161 LibCrypto:
162 TestAES.cpp
163 TestASN1.cpp
164 TestBigInteger.cpp
165 TestChaCha20.cpp
166 TestChacha20Poly1305.cpp
167 TestChecksum.cpp
168 TestCurves.cpp
169 TestEd25519.cpp
170 TestHMAC.cpp
171 TestHash.cpp
172 TestPBKDF2.cpp
N… todo_tests.txt
:se nowrap

Bitmap

Bitmap

```
18
19 TEST_CASE(find_first_set)
20 {
21     auto bitmap = MUST(Bitmap::create(128, false));
22     bitmap.set(69, true);
23     EXPECT_EQ(bitmap.find_first_set().value(), 69u);
24 }
25
```

Bitmap

```
18
19 TEST_CASE(find_first_set)
20 {
21     auto bitmap = MUST(Bitmap::create(128, false));
22     bitmap.set(69, true);
23     EXPECT_EQ(bitmap.find_first_set().value(), 69u);
24 }
25
```

```
{
    auto bitmap = MUST(Bitmap::create(168, false));
    bitmap.set(34, true);
    bitmap.set(97, true);
```

Bitmap

```
18
19 TEST_CASE(find_first_set)
20 {
21     auto bitmap = MUST(Bitmap::create(128, false));
22     bitmap.set(69, true);
23     EXPECT_EQ(bitmap.find_first_set().value(), 69u);
24 }
25
```

```
{
    auto bitmap = MUST(Bitmap::create(288, false));
    bitmap.set_range(48, 32, true);
    bitmap.set_range(94, 39, true);
    bitmap.set_range(190, 71, true);
    bitmap.set_range(190 + 71 - 7, 21, false); // slightly overlapping clear
```

```
{
    auto bitmap = MUST(Bitmap::create(168, false));
    bitmap.set(34, true);
    bitmap.set(97, true);
```

Bitmap

```
18
19 TEST_CASE(find_first_set)
20 {
21     auto bitmap = MUST(Bitmap::create(128, false));
22     bitmap.set(69, true);
23     EXPECT_EQ(bitmap.find_first_set().value(), 69u);
24 }
25
```

```
{
    auto bitmap = MUST(Bitmap::create(288, false));
    bitmap.set_range(48, 32, true);
    bitmap.set_range(94, 39, true);
    bitmap.set_range(190, 71, true);
    bitmap.set_range(190 + 71 - 7, 21, false); // slightly overlapping clear
```

```
{
    auto bitmap = MUST(Bitmap::create(168, false));
    bitmap.set(34, true);
    bitmap.set(97, true);
```

```
{
    auto bitmap = MUST(Bitmap::create(128 + 24, false));
    bitmap.set(34, true);
    bitmap.set(126, true);
```

```
363 RANDOMIZED_TEST_CASE(find_first)
364 {
365     GEN(init, Gen::boolean());
366     GEN(size, Gen::number_u64(1, 64));
367
368     GEN(new_value, Gen::boolean());
369     GEN(i, Gen::number_u64(size - 1));
370
371     auto bitmap = MUST(Bitmap::create(size, init));
372     bitmap.set(i, new_value);
373
374     Optional<size_t> result = new_value
375         ? bitmap.find_first_set()
376         : bitmap.find_first_unset();
377
378     auto expected_found_index = init == new_value ? 0 : i;
379     EXPECT_EQ(result.value(), expected_found_index);
380 }
```

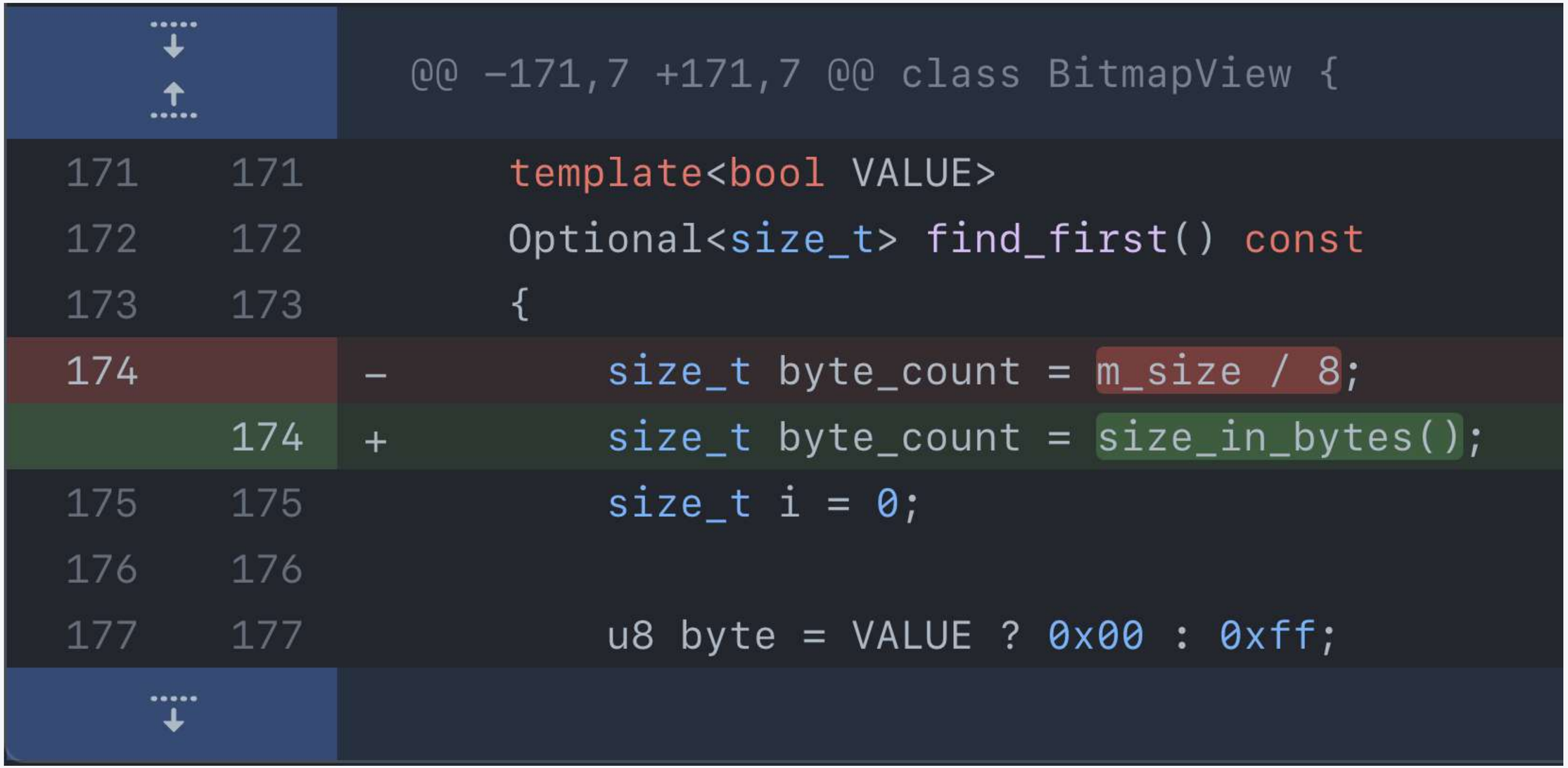
```
363 RANDOMIZED_TEST_CASE(find_first)
364 {
365     GEN(init, Gen::boolean());
366     GEN(size, Gen::number_u64(1, 64));
367
368     GEN(new_value, Gen::boolean());
369     GEN(i, Gen::number_u64(size - 1));
370
371     auto bitmap = MUST(Bitmap::create(size, init));
372     bitmap.set(i, new_value);
373
374
375
376
377
378
379
380 }
```

```
Running test 'find_first'.
init = false
size = 1
new_value = false
i = 0
FAIL: /Users/martin/Localhost/cloned/serenity/Tests/AK/TestBitmap.cpp:370: EXPECT(result.has_value()) failed
Failed test 'find_first' in 0ms
```

```
[[nodiscard]] size_t size_in_bytes() const {
    return ceil_div(m_size, static_cast<size_t>(8));
}
```

```
[[nodiscard]] size_t size_in_bytes() const {
    return ceil_div(m_size, static_cast<size_t>(8));
}
```

```
template<bool VALUE>
Optional<size_t> find_first() const
{
    size_t byte_count = m_size / 8;
    size_t i = 0;
```


@@ -171,7 +171,7 @@ class BitmapView {
171 171 template<bool VALUE>
172 172 Optional<size_t> find_first() const
173 173 {
174 - size_t byte_count = m_size / 8;
174 + size_t byte_count = size_in_bytes();
175 175 size_t i = 0;
176 176
177 177 u8 byte = VALUE ? 0x00 : 0xff;

AK: Fix one-off error in BitmapView::find_first and find_one_anywhere #21409

Edit | Code

Merged **timschumi** merged 1 commit into `SerenityOS:master` from `Janiczek:fix-bitmap` on Oct 11, 2023

Conversation 0 | Commits 1 | Checks 15 | Files changed 2 | +21 −3

Janiczek commented on Oct 11, 2023 • edited — Contributor

The mentioned functions used `m_size / 8` instead of `size_in_bytes()` (division with ceiling rounding mode), which resulted in an off-by-one error such that the functions didn't search in the last not-fully-8-bits byte.

Using `size_in_bytes()` instead of `m_size / 8` fixes this.

> **Note**
>
> This was found using `RANDOMIZED_TEST_CASE` (PR ready for review). I'd appreciate reviews there as well, so that I could commit the randomized tests that found the issue as well!

Reviewers

 gmta ✓

 timschumi ✓

Assignees

No one assigned

Labels

None yet

Projects

vim

```
38 TEST_CASE(Complex)
39 {
40     auto a = Complex<float> { 1.f, 1.f };
41     auto b = complex_real_unit<double> + Complex<double> { 0, 1 } * 1;
42     EXPECT_APPROXIMATE(a.real(), b.real());
43     EXPECT_APPROXIMATE(a.imag(), b.imag());
44
45 #ifdef AKCOMPLEX_CAN_USE_MATH_H
46     EXPECT_APPROXIMATE((complex_imag_unit<float> - complex_imag_unit<float>).magnitude(), 0);
47     EXPECT_APPROXIMATE((complex_imag_unit<float> + complex_real_unit<float>).magnitude(), sqrt(2));
48
49     auto c = Complex<double> { 0., 1. };
50     auto d = Complex<double>::from_polar(1., M_PI / 2.);
51     EXPECT_APPROXIMATE(c.real(), d.real());
52     EXPECT_APPROXIMATE(c.imag(), d.imag());
53
54     c = Complex<double> { -1., 1. };
55     d = Complex<double>::from_polar(sqrt(2.), 3. * M_PI / 4.);
56     EXPECT_APPROXIMATE(c.real(), d.real());
57     EXPECT_APPROXIMATE(c.imag(), d.imag());
58     EXPECT_APPROXIMATE(d.phase(), 3. * M_PI / 4.);
59     EXPECT_APPROXIMATE(c.magnitude(), d.magnitude());
60     EXPECT_APPROXIMATE(c.magnitude(), sqrt(2.));
61 #endif
62     EXPECT_EQ((complex_imag_unit<double> * complex_imag_unit<double>).real(), -1.);
63     EXPECT_EQ((complex_imag_unit<double> / complex_imag_unit<double>).real(), 1.);
64
65     EXPECT_EQ(Complex(1., 10.) == (Complex<double>(1., 0.) + Complex(0., 10.)), true);
66     EXPECT_EQ(Complex(1., 10.) != (Complex<double>(1., 1.) + Complex(0., 10.)), true);
67 #ifdef AKCOMPLEX_CAN_USE_MATH_H
68     EXPECT_EQ(approx_eq(Complex<int>(1), Complex<float>(1.0000004f)), true);
69     EXPECT_APPROXIMATE(cexp(Complex<double>(0., 1.) * M_PI).real(), -1.);
70 #endif
71 }
72
```

NORMAL pbt-complex! Tests/AK/TestComplex.cpp cpp utf-8[unix] 21% ln :55/261 ≡%:1

```
In file included from /Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:9:
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:86:20: error: member reference base type 'const int' is not a stru
cture or union
   86 |         m_real += x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:77:11: note: in instantiation of function template specialization 'AK::
Complex<double>::operator+=<int>' requested here
   77 |         c += 1;
      |           ^
In file included from /Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:9:
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:101:20: error: member reference base type 'const int' is not a str
ucture or union
  101 |         m_real -= x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:82:11: note: in instantiation of function template specialization 'AK::
Complex<double>::operator-=<int>' requested here
   82 |         c -= 1;
      |           ^
In file included from /Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:9:
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:86:20: error: member reference base type 'const double' is not a s
tructure or union
   86 |         m_real += x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:152:11: note: in instantiation of function template specialization
 'AK::Complex<double>::operator+=<double>' requested here
  152 |         x += a;
      |           ^
/Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:166:18: note: in instantiation of function template specialization 'AK:
:Complex<double>::operator+<double>' requested here
  166 |     auto c2 = c1 + r2;
      |                  ^
In file included from /Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:9:
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:101:20: error: member reference base type 'const double' is not a
structure or union
  101 |         m_real -= x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:168:11: note: in instantiation of function template specialization
```

-bash

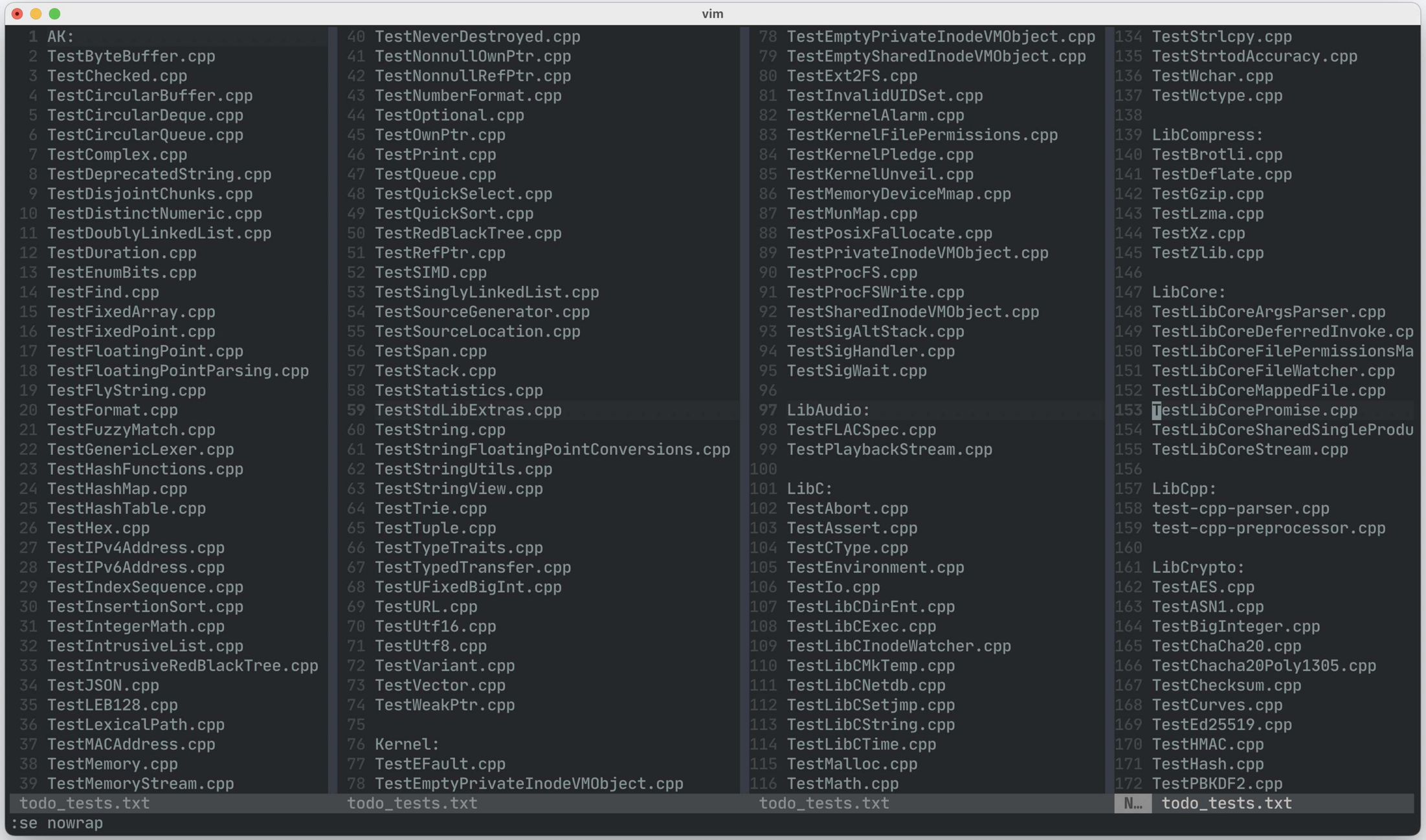
```
In file included from /Users/martin/Localhost/cloned/serenity/Tests/AK/TestComplex.cpp:9:
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:86:20: error: member reference base type 'const int' is not a stru
cture or union
   86 |         m_real += x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/sere                                                    tion template specialization 'AK::
Complex<double>::operator+=<int>' r
   77 |         c += 1;
      |           ^
In file included from /Users/martin
/Users/martin/Localhost/cloned/sere                                                    base type 'const int' is not a str
ucture or union
  101 |         m_real -= x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/sere                                                    tion template specialization 'AK::
Complex<double>::operator-=<int>' r
   82 |         c -= 1;
      |           ^
In file included from /Users/martin
/Users/martin/Localhost/cloned/sere                                                    ase type 'const double' is not a s
tructure or union
   86 |         m_real += x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/sere                                                    f function template specialization
 'AK::Complex<double>::operator+=<d
  152 |         x += a;
      |           ^
/Users/martin/Localhost/cloned/sere                                                    ction template specialization 'AK:
:Complex<double>::operator+<double>
  166 |     auto c2 = c1 + r2;
      |                    ^
In file included from /Users/martin
/Users/martin/Localhost/cloned/sere                                                    base type 'const double' is not a
structure or union
  101 |         m_real -= x.real();
      |                   ~^~~~~
/Users/martin/Localhost/cloned/serenity/Meta/Lagom/../../AK/Complex.h:168:11: note: in instantiation of function template specialization
```

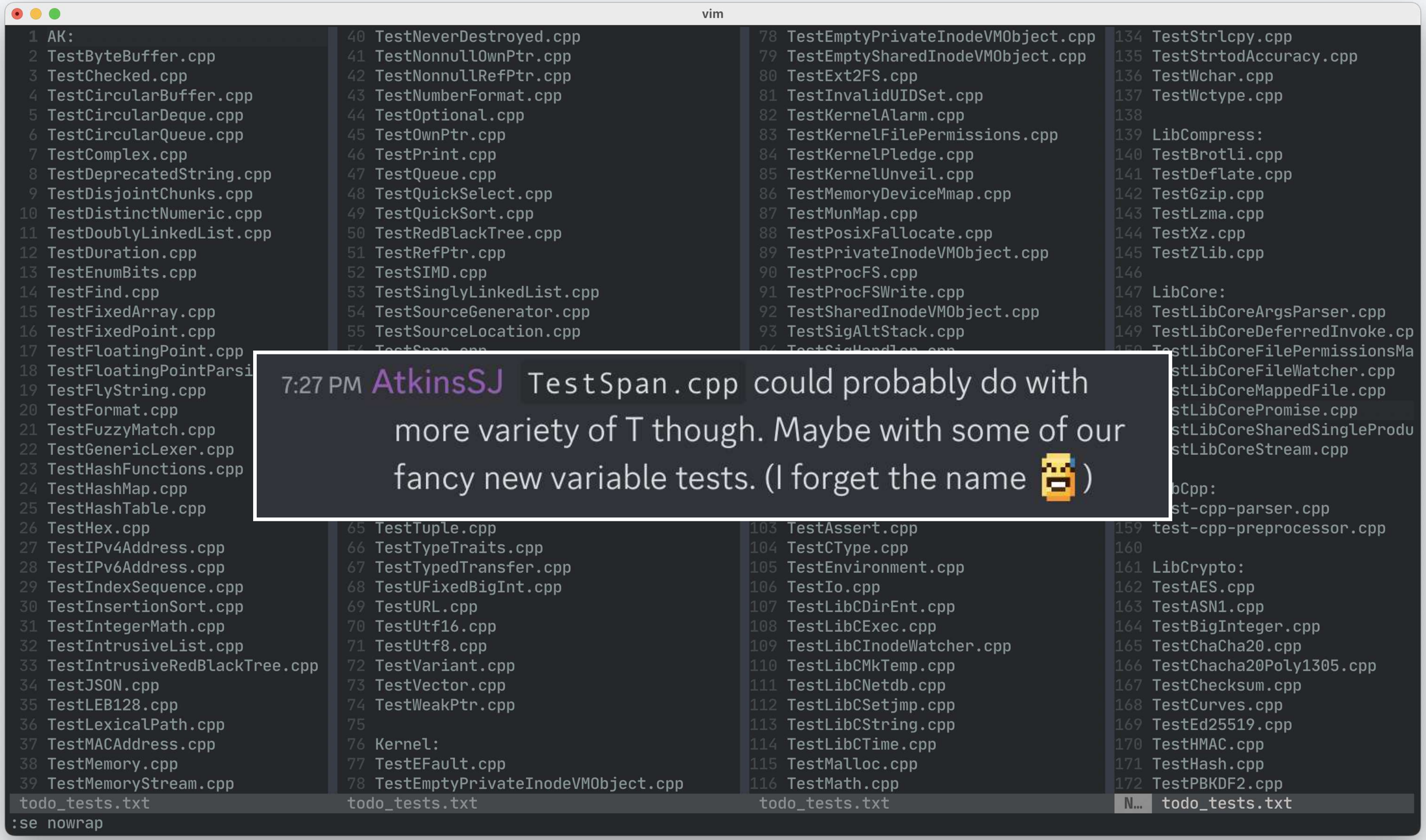
12 AK/Complex.h

```
@@ -83,7 +83,7 @@ class [[gnu::packed]] Complex {
83  83      template<AK::Concepts::Arithmetic U>
84  84      constexpr Complex<T> operator+=(U const& x)
85  85      {
86      -       m_real += x.real();
    86  +       m_real += x;
87  87          return *this;
88  88      }
89  89
@@ -98,7 +98,7 @@ class [[gnu::packed]] Complex {
98  98      template<AK::Concepts::Arithmetic U>
99  99      constexpr Complex<T> operator-=(U const& x)
100 100     {
101     -       m_real -= x.real();
    101 +       m_real -= x;
102 102         return *this;
103 103     }
104 104
@@ -224,31 +224,31 @@ class [[gnu::packed]] Complex {
```

```
233 233
234 234   template<AK::Concepts::Arithmetic T, AK::Concepts::Arithmetic U>
235     - constexpr Complex<T> operator-(U const& b, Complex<T> const& a)
    235 + constexpr Complex<T> operator-(U const& a, Complex<T> const& b)
236 236   {
237 237       Complex<T> x = a;
238 238       x -= b;
239 239       return x;
240 240   }
241 241
```

```
72
73 TEST_CASE(real_operators_regression)
74 {
75     {
76         auto c1 = Complex(1., 1.);
77         auto c2 = 1 - c1;
78         EXPECT_EQ(c2.real(), 0);
79         EXPECT_EQ(c2.imag(), -1);
80     }
81     {
82         auto c1 = Complex(1., 1.);
83         auto c2 = 1 / c1;
84         EXPECT_EQ(c2.real(), 0.5);
85         EXPECT_EQ(c2.imag(), -0.5);
86     }
87 }
88
```


vim
1 AK:
2 TestByteBuffer.cpp
3 TestChecked.cpp
4 TestCircularBuffer.cpp
5 TestCircularDeque.cpp
6 TestCircularQueue.cpp
7 TestComplex.cpp
8 TestDeprecatedString.cpp
9 TestDisjointChunks.cpp
10 TestDistinctNumeric.cpp
11 TestDoublyLinkedList.cpp
12 TestDuration.cpp
13 TestEnumBits.cpp
14 TestFind.cpp
15 TestFixedArray.cpp
16 TestFixedPoint.cpp
17 TestFloatingPoint.cpp
18 TestFloatingPointParsing.cpp
19 TestFlyString.cpp
20 TestFormat.cpp
21 TestFuzzyMatch.cpp
22 TestGenericLexer.cpp
23 TestHashFunctions.cpp
24 TestHashMap.cpp
25 TestHashTable.cpp
26 TestHex.cpp
27 TestIPv4Address.cpp
28 TestIPv6Address.cpp
29 TestIndexSequence.cpp
30 TestInsertionSort.cpp
31 TestIntegerMath.cpp
32 TestIntrusiveList.cpp
33 TestIntrusiveRedBlackTree.cpp
34 TestJSON.cpp
35 TestLEB128.cpp
36 TestLexicalPath.cpp
37 TestMACAddress.cpp
38 TestMemory.cpp
39 TestMemoryStream.cpp
todo_tests.txt
40 TestNeverDestroyed.cpp
41 TestNonnullOwnPtr.cpp
42 TestNonnullRefPtr.cpp
43 TestNumberFormat.cpp
44 TestOptional.cpp
45 TestOwnPtr.cpp
46 TestPrint.cpp
47 TestQueue.cpp
48 TestQuickSelect.cpp
49 TestQuickSort.cpp
50 TestRedBlackTree.cpp
51 TestRefPtr.cpp
52 TestSIMD.cpp
53 TestSinglyLinkedList.cpp
54 TestSourceGenerator.cpp
55 TestSourceLocation.cpp
56 TestSpan.cpp
57 TestStack.cpp
58 TestStatistics.cpp
59 TestStdLibExtras.cpp
60 TestString.cpp
61 TestStringFloatingPointConversions.cpp
62 TestStringUtils.cpp
63 TestStringView.cpp
64 TestTrie.cpp
65 TestTuple.cpp
66 TestTypeTraits.cpp
67 TestTypedTransfer.cpp
68 TestUFixedBigInt.cpp
69 TestURL.cpp
70 TestUtf16.cpp
71 TestUtf8.cpp
72 TestVariant.cpp
73 TestVector.cpp
74 TestWeakPtr.cpp
75
76 Kernel:
77 TestEFault.cpp
78 TestEmptyPrivateInodeVMObject.cpp
todo_tests.txt
78 TestEmptyPrivateInodeVMObject.cpp
79 TestEmptySharedInodeVMObject.cpp
80 TestExt2FS.cpp
81 TestInvalidUIDSet.cpp
82 TestKernelAlarm.cpp
83 TestKernelFilePermissions.cpp
84 TestKernelPledge.cpp
85 TestKernelUnveil.cpp
86 TestMemoryDeviceMmap.cpp
87 TestMunMap.cpp
88 TestPosixFallocate.cpp
89 TestPrivateInodeVMObject.cpp
90 TestProcFS.cpp
91 TestProcFSWrite.cpp
92 TestSharedInodeVMObject.cpp
93 TestSigAltStack.cpp
94 TestSigHandler.cpp
95 TestSigWait.cpp
96
97 LibAudio:
98 TestFLACSpec.cpp
99 TestPlaybackStream.cpp
100
101 LibC:
102 TestAbort.cpp
103 TestAssert.cpp
104 TestCType.cpp
105 TestEnvironment.cpp
106 TestIo.cpp
107 TestLibCDirEnt.cpp
108 TestLibCExec.cpp
109 TestLibCInodeWatcher.cpp
110 TestLibCMkTemp.cpp
111 TestLibCNetdb.cpp
112 TestLibCSetjmp.cpp
113 TestLibCString.cpp
114 TestLibCTime.cpp
115 TestMalloc.cpp
116 TestMath.cpp
todo_tests.txt
134 TestStrlcpy.cpp
135 TestStrtodAccuracy.cpp
136 TestWchar.cpp
137 TestWctype.cpp
138
139 LibCompress:
140 TestBrotli.cpp
141 TestDeflate.cpp
142 TestGzip.cpp
143 TestLzma.cpp
144 TestXz.cpp
145 TestZlib.cpp
146
147 LibCore:
148 TestLibCoreArgsParser.cpp
149 TestLibCoreDeferredInvoke.cp
150 TestLibCoreFilePermissionsMa
151 TestLibCoreFileWatcher.cpp
152 TestLibCoreMappedFile.cpp
153 TestLibCorePromise.cpp
154 TestLibCoreSharedSingleProdu
155 TestLibCoreStream.cpp
156
157 LibCpp:
158 test-cpp-parser.cpp
159 test-cpp-preprocessor.cpp
160
161 LibCrypto:
162 TestAES.cpp
163 TestASN1.cpp
164 TestBigInteger.cpp
165 TestChaCha20.cpp
166 TestChacha20Poly1305.cpp
167 TestChecksum.cpp
168 TestCurves.cpp
169 TestEd25519.cpp
170 TestHMAC.cpp
171 TestHash.cpp
172 TestPBKDF2.cpp
N… todo_tests.txt
:se nowrap


vim
AK:
TestByteBuffer.cpp
TestChecked.cpp
TestCircularBuffer.cpp
TestCircularDeque.cpp
TestCircularQueue.cpp
TestComplex.cpp
TestDeprecatedString.cpp
TestDisjointChunks.cpp
TestDistinctNumeric.cpp
TestDoublyLinkedList.cpp
TestDuration.cpp
TestEnumBits.cpp
TestFind.cpp
TestFixedArray.cpp
TestFixedPoint.cpp
TestFloatingPoint.cpp
TestFloatingPointParsi
TestFlyString.cpp
TestFormat.cpp
TestFuzzyMatch.cpp
TestGenericLexer.cpp
TestHashFunctions.cpp
TestHashMap.cpp
TestHashTable.cpp
TestHex.cpp
TestIPv4Address.cpp
TestIPv6Address.cpp
TestIndexSequence.cpp
TestInsertionSort.cpp
TestIntegerMath.cpp
TestIntrusiveList.cpp
TestIntrusiveRedBlackTree.cpp
TestJSON.cpp
TestLEB128.cpp
TestLexicalPath.cpp
TestMACAddress.cpp
TestMemory.cpp
TestMemoryStream.cpp
TestNeverDestroyed.cpp
TestNonnullOwnPtr.cpp
TestNonnullRefPtr.cpp
TestNumberFormat.cpp
TestOptional.cpp
TestOwnPtr.cpp
TestPrint.cpp
TestQueue.cpp
TestQuickSelect.cpp
TestQuickSort.cpp
TestRedBlackTree.cpp
TestRefPtr.cpp
TestSIMD.cpp
TestSinglyLinkedList.cpp
TestSourceGenerator.cpp
TestSourceLocation.cpp
TestTuple.cpp
TestTypeTraits.cpp
TestTypedTransfer.cpp
TestUFixedBigInt.cpp
TestURL.cpp
TestUtf16.cpp
TestUtf8.cpp
TestVariant.cpp
TestVector.cpp
TestWeakPtr.cpp
Kernel:
TestEFault.cpp
TestEmptyPrivateInodeVMObject.cpp
TestEmptyPrivateInodeVMObject.cpp
TestEmptySharedInodeVMObject.cpp
TestExt2FS.cpp
TestInvalidUIDSet.cpp
TestKernelAlarm.cpp
TestKernelFilePermissions.cpp
TestKernelPledge.cpp
TestKernelUnveil.cpp
TestMemoryDeviceMmap.cpp
TestMunMap.cpp
TestPosixFallocate.cpp
TestPrivateInodeVMObject.cpp
TestProcFS.cpp
TestProcFSWrite.cpp
TestSharedInodeVMObject.cpp
TestSigAltStack.cpp
TestAssert.cpp
TestCType.cpp
TestEnvironment.cpp
TestIo.cpp
TestLibCDirEnt.cpp
TestLibCExec.cpp
TestLibCInodeWatcher.cpp
TestLibCMkTemp.cpp
TestLibCNetdb.cpp
TestLibCSetjmp.cpp
TestLibCString.cpp
TestLibCTime.cpp
TestMalloc.cpp
TestMath.cpp
TestStrlcpy.cpp
TestStrtodAccuracy.cpp
TestWchar.cpp
TestWctype.cpp
LibCompress:
TestBrotli.cpp
TestDeflate.cpp
TestGzip.cpp
TestLzma.cpp
TestXz.cpp
TestZlib.cpp
LibCore:
TestLibCoreArgsParser.cpp
TestLibCoreDeferredInvoke.cp
stLibCoreFileWatcher.cpp
stLibCoreMappedFile.cpp
stLibCorePromise.cpp
stLibCoreSharedSingleProdu
stLibCoreStream.cpp
bCpp:
st-cpp-parser.cpp
test-cpp-preprocessor.cpp
LibCrypto:
TestAES.cpp
TestASN1.cpp
TestBigInteger.cpp
TestChaCha20.cpp
TestChacha20Poly1305.cpp
TestChecksum.cpp
TestCurves.cpp
TestEd25519.cpp
TestHMAC.cpp
TestHash.cpp
TestPBKDF2.cpp
7:27 PM AtkinsSJ TestSpan.cpp could probably do with more variety of T though. Maybe with some of our fancy new variable tests. (I forget the name)
todo_tests.txt
todo_tests.txt
todo_tests.txt
N… todo_tests.txt
:se nowrap

Summary

Summary

- Property tests are great!

Summary

- Property tests are great!
- For the love of God, steal from Hypothesis

Summary

- Property tests are great!
- For the love of God, steal from Hypothesis
- Functional C++ is hard; don’t go against the grain

Summary

- Property tests are great!
- For the love of God, steal from Hypothesis
- Functional C++ is hard; don’t go against the grain
- SerenityOS+PBT is a learning opportunity

Thank you!

@janiczek