Mutation Testing with C++

Using C++ Mutation Test Environment.

Mutate++

https://github.com/nlohmann/mutate_cpp

Prepared by: Sergio Garcia & Jose Pastor

What is mutation testing?

- Is a method for improving software tests and consecuently improve the software
- Mutation testing involves modifying (mutating) a program in small ways.
- If a test does not fail when testing a mutated software it has detected a problem in the test

Comercial Products

- **PlexTest**: http://www.itregister.com.au/products/plextest_detail.htm
- **Insure++**: http://www.parasoft.com/jsp/products/insure.jsp;jsessionid=baacpvbaDywLID?itemId=63
- **MILU** (may be only for C): http://www.dcs.kcl.ac.uk/pg/jiayue/milu/

Example of a SUT

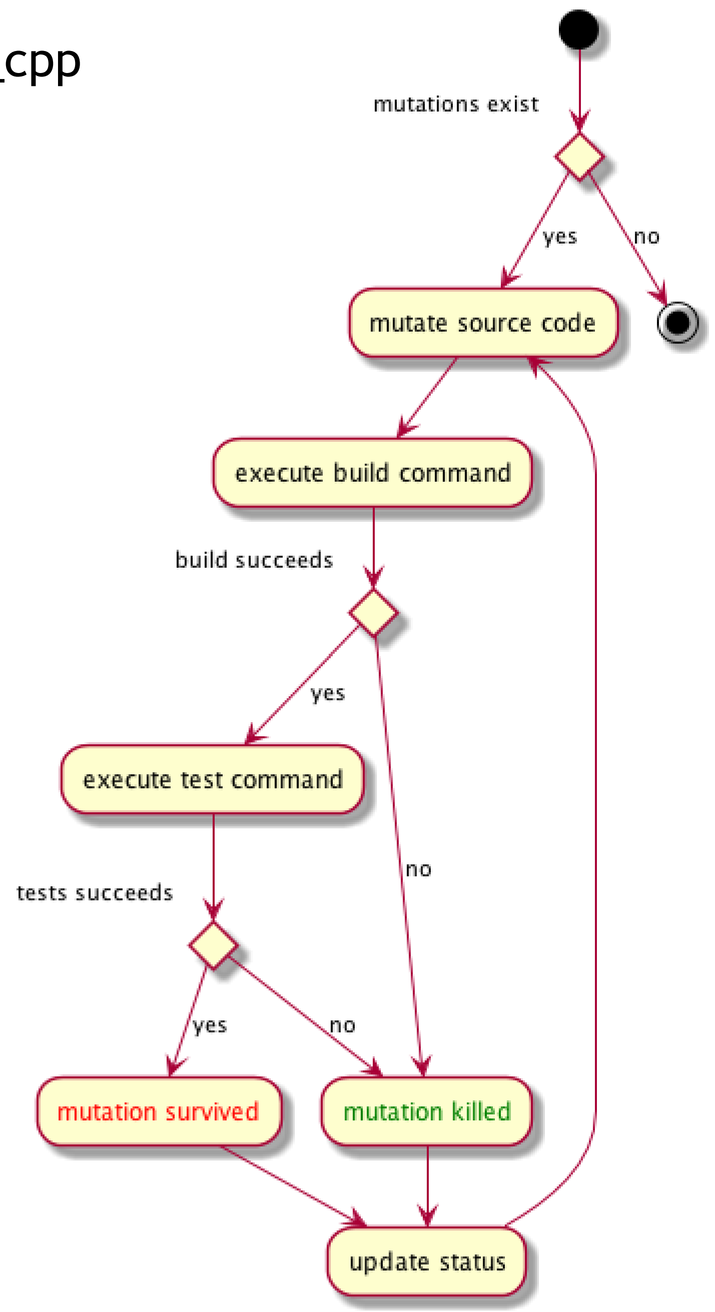
```
#include "example.h"

double add_numbers(const double f1, const double f2)
{
    return f1 + f2;
}

double subtract_numbers(const double f1, const double f2)
{
    return f1 - f2;
}

double multiply_numbers(const double f1, const double f2)
{
    return f1*f2;
}
```

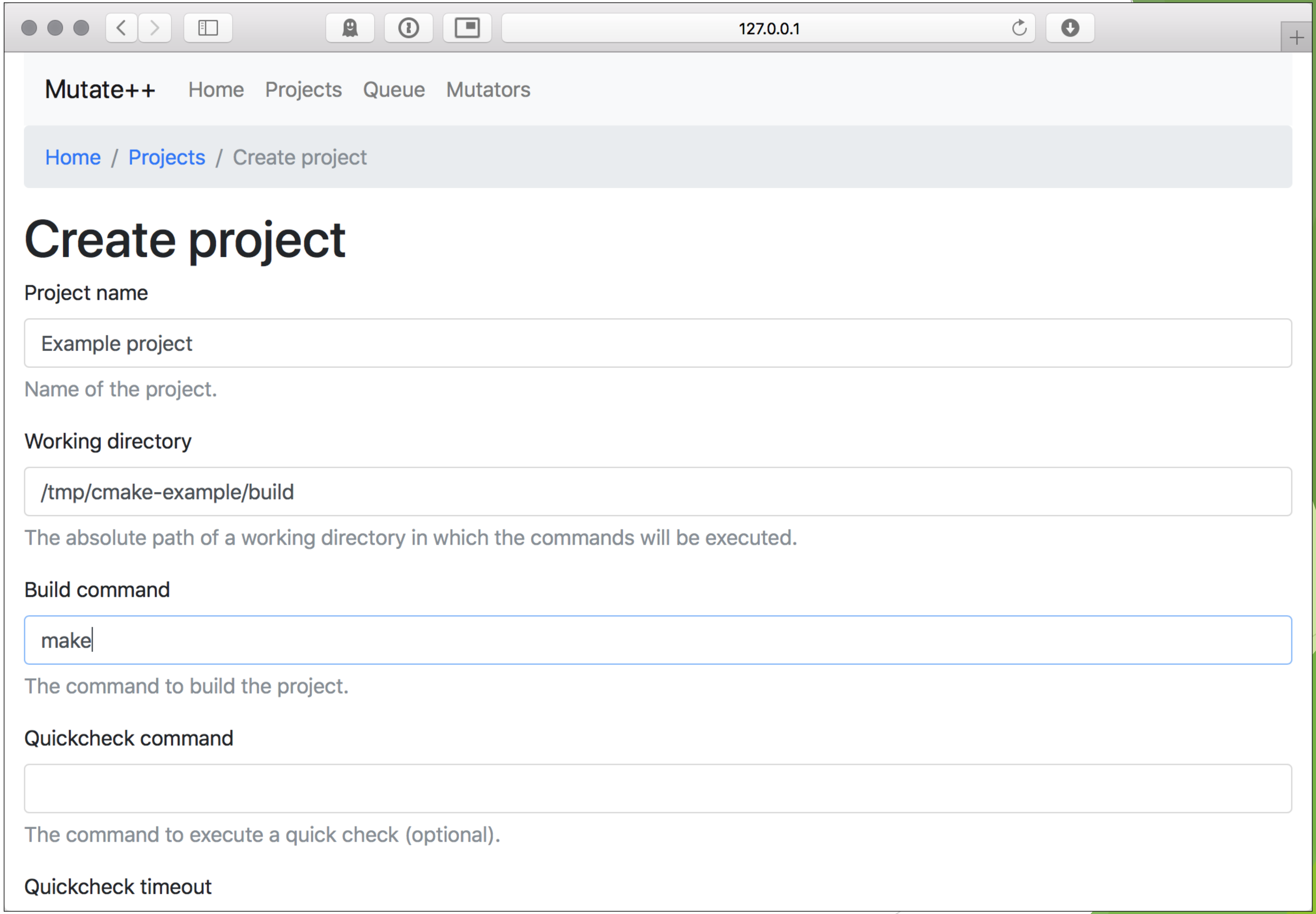
Example of Test

```
#include "gtest/gtest.h"
#include "example.h"

TEST(example, add)
{
    double res;
    res = add_numbers(1.0, 2.0);
    ASSERT_NEAR(res, 3.0, 1.0e-11);
}
TEST(example, add)
{
    double res;
    res = add_numbers(1.0, 2.0);
    ASSERT_NEAR(res, 3.0, 1.0e-11);
}
```

NOTE: Multiply operation not tested

https://github.com/nlohmann/mutate_cpp





Create project

Project name

Example project

Name of the project.

Working directory

/tmp/cmake-example/build

The absolute path of a working directory in which the commands will be executed.

Build command

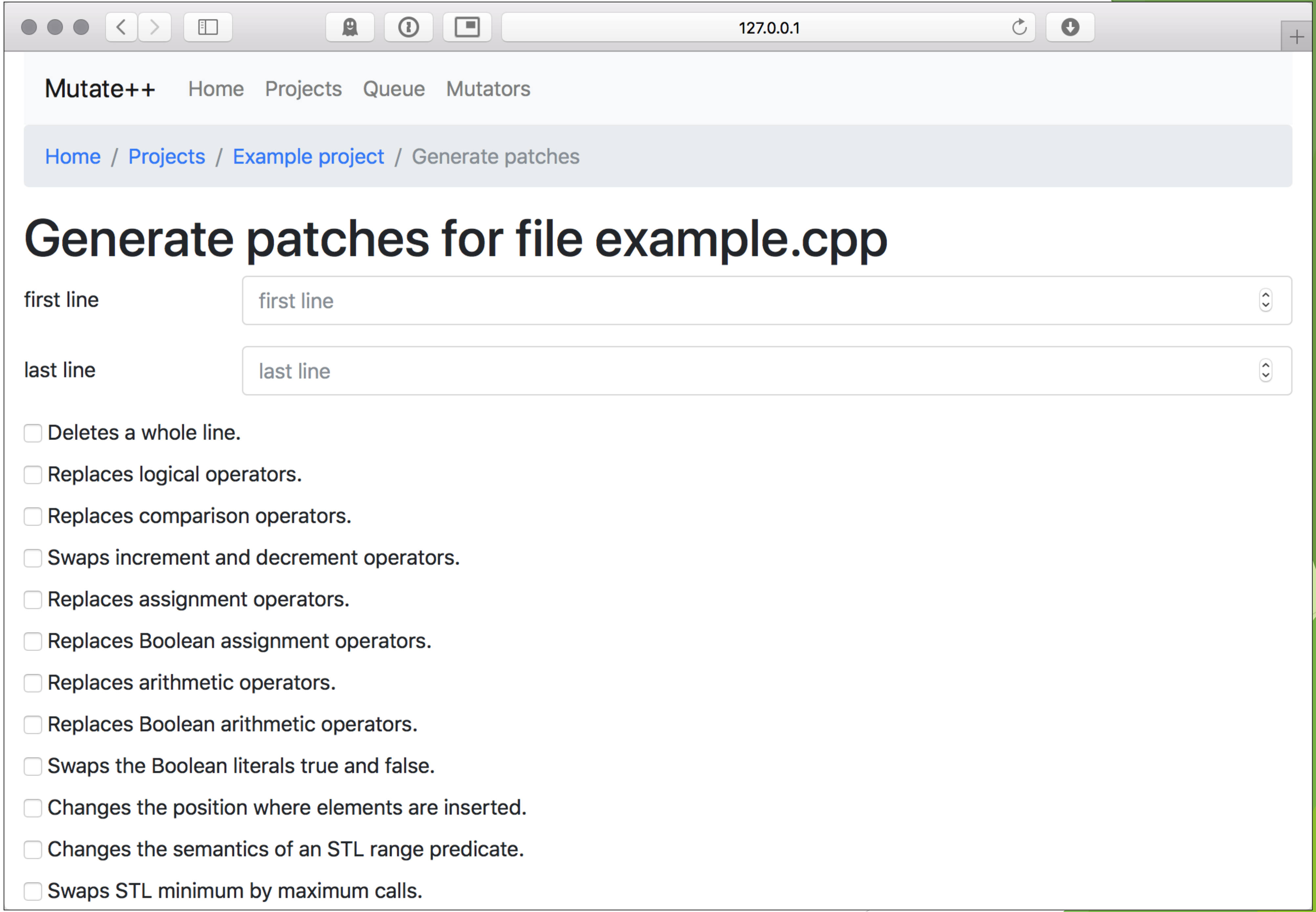
make

The command to build the project.

Quickcheck command

The command to execute a quick check (optional).

Quickcheck timeout



Mutate++ Home Projects Queue Mutators

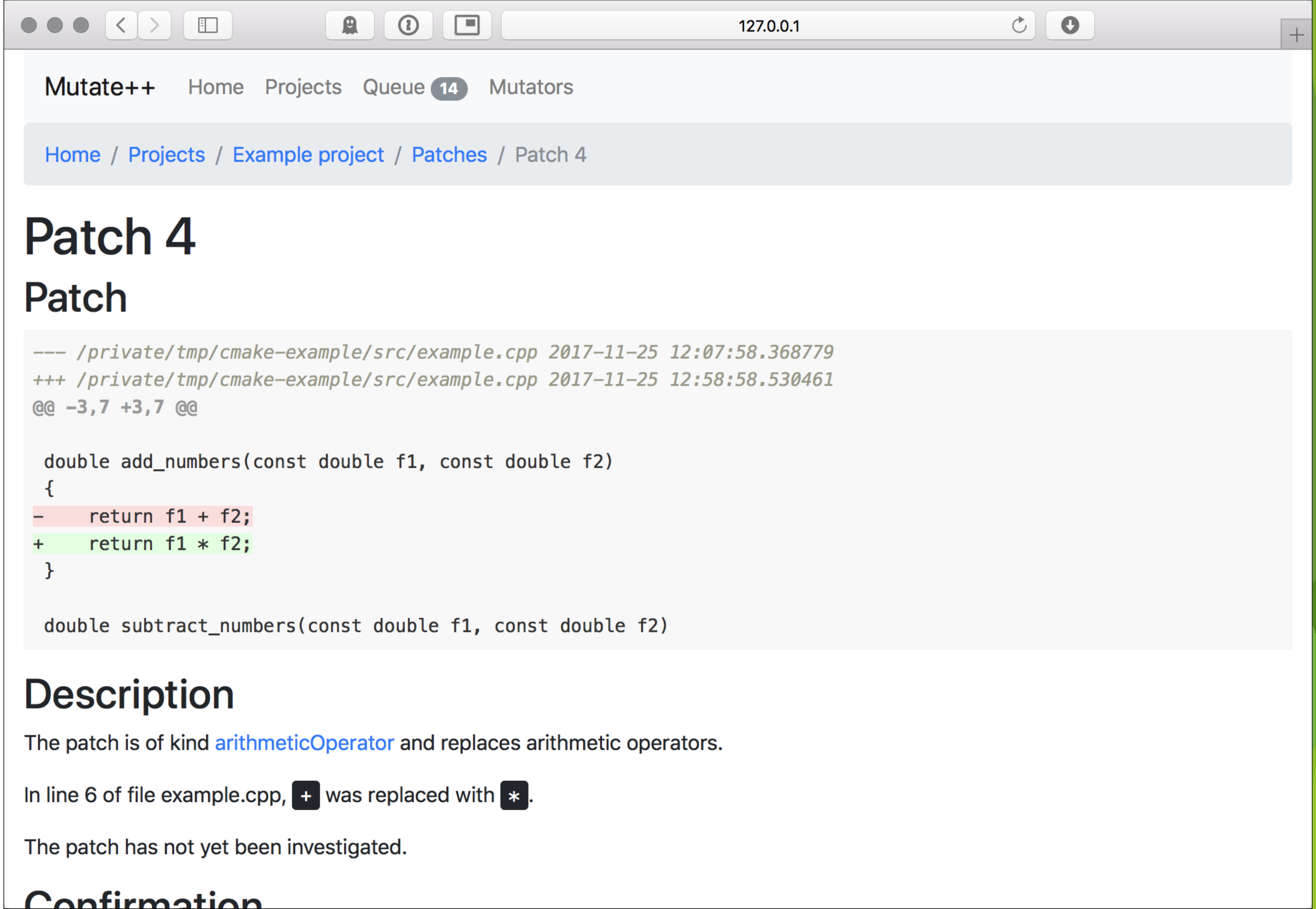
Home / Projects / Example project / Generate patches

Generate patches for file example.cpp

first line: first line

last line: last line

- [ ] Deletes a whole line.
- [ ] Replaces logical operators.
- [ ] Replaces comparison operators.
- [ ] Swaps increment and decrement operators.
- [ ] Replaces assignment operators.
- [ ] Replaces Boolean assignment operators.
- [ ] Replaces arithmetic operators.
- [ ] Replaces Boolean arithmetic operators.
- [ ] Swaps the Boolean literals true and false.
- [ ] Changes the position where elements are inserted.
- [ ] Changes the semantics of an STL range predicate.
- [ ] Swaps STL minimum by maximum calls.



Patch 4

Patch

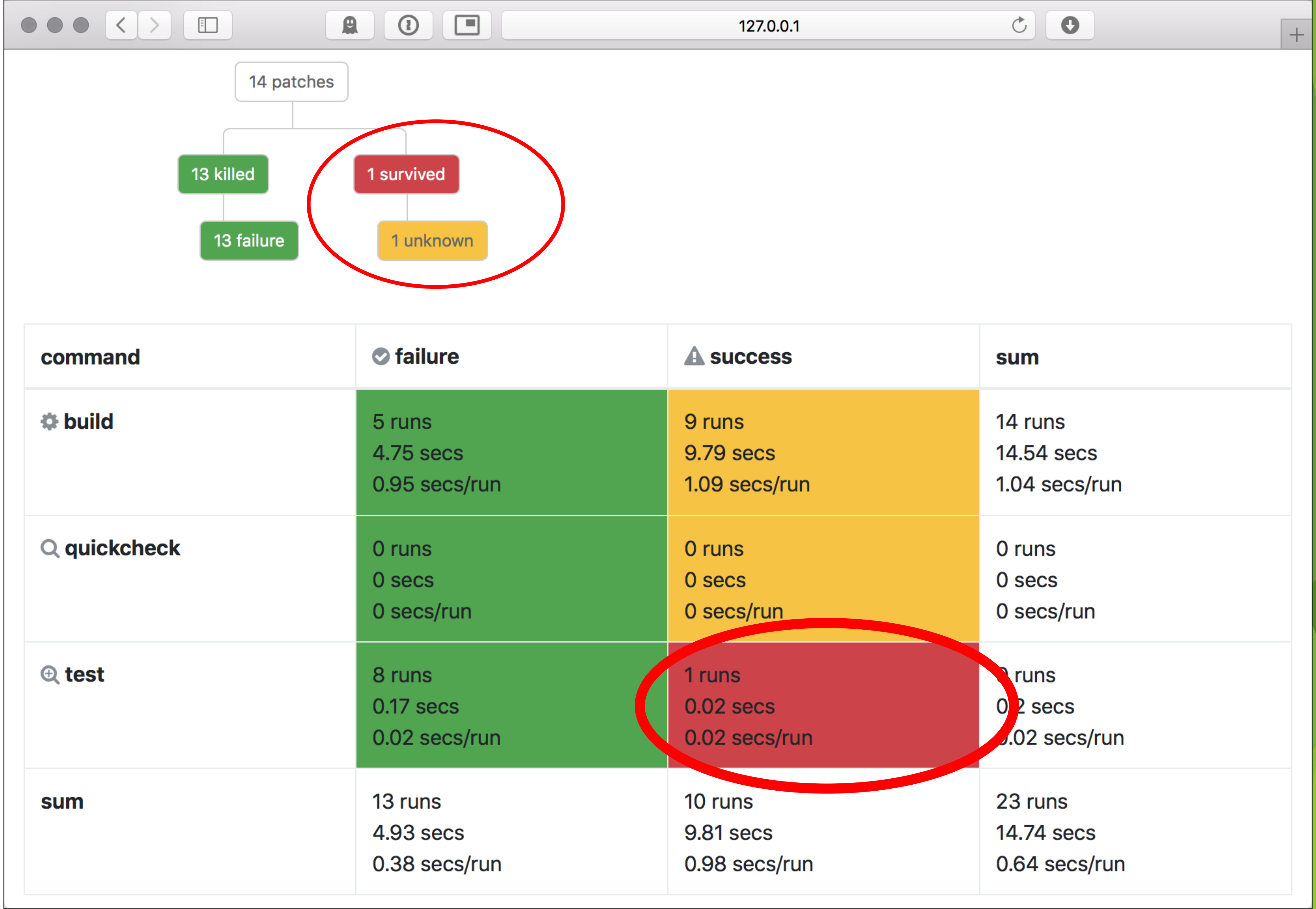
```
--- /private/tmp/cmake-example/src/example.cpp 2017-11-25 12:07:58.368779
+++ /private/tmp/cmake-example/src/example.cpp 2017-11-25 12:58:58.530461
@@ -3,7 +3,7 @@

 double add_numbers(const double f1, const double f2)
 {
-    return f1 + f2;
+    return f1 * f2;
 }

 double subtract_numbers(const double f1, const double f2)
```

Description

The patch is of kind arithmeticOperator and replaces arithmetic operators.

In line 6 of file example.cpp, `+` was replaced with `*`.

The patch has not yet been investigated.



14 patches
- 13 killed
  - 13 failure
- 1 survived
  - 1 unknown

command	failure	success	sum
build	5 runs 4.75 secs 0.95 secs/run	9 runs 9.79 secs 1.09 secs/run	14 runs 14.54 secs 1.04 secs/run
quickcheck	0 runs 0 secs 0 secs/run	0 runs 0 secs 0 secs/run	0 runs 0 secs 0 secs/run
test	8 runs 0.17 secs 0.02 secs/run	1 runs 0.02 secs 0.02 secs/run	9 runs 0.2 secs 0.02 secs/run
sum	13 runs 4.93 secs 0.38 secs/run	10 runs 9.81 secs 0.98 secs/run	23 runs 14.74 secs 0.64 secs/run