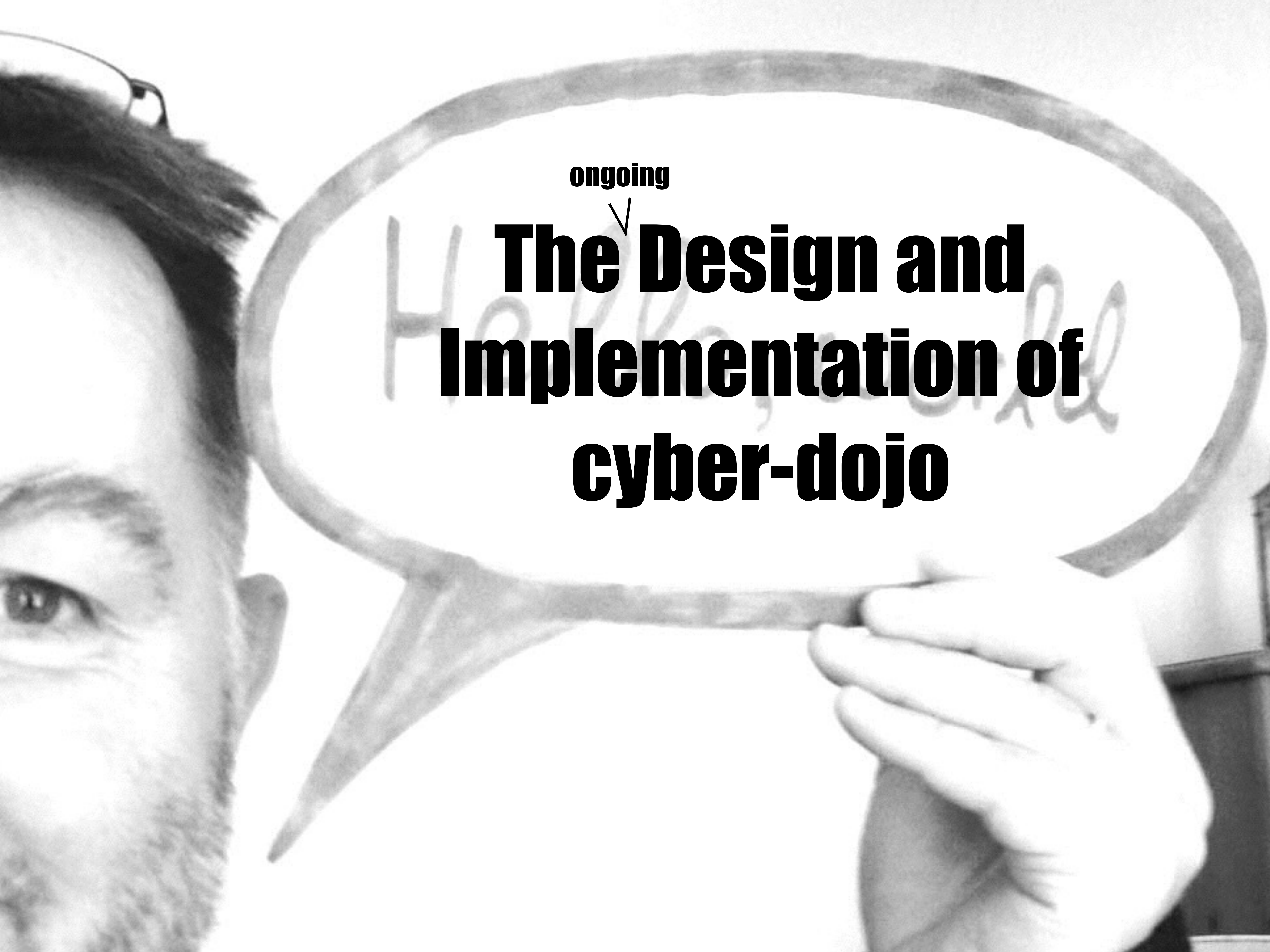


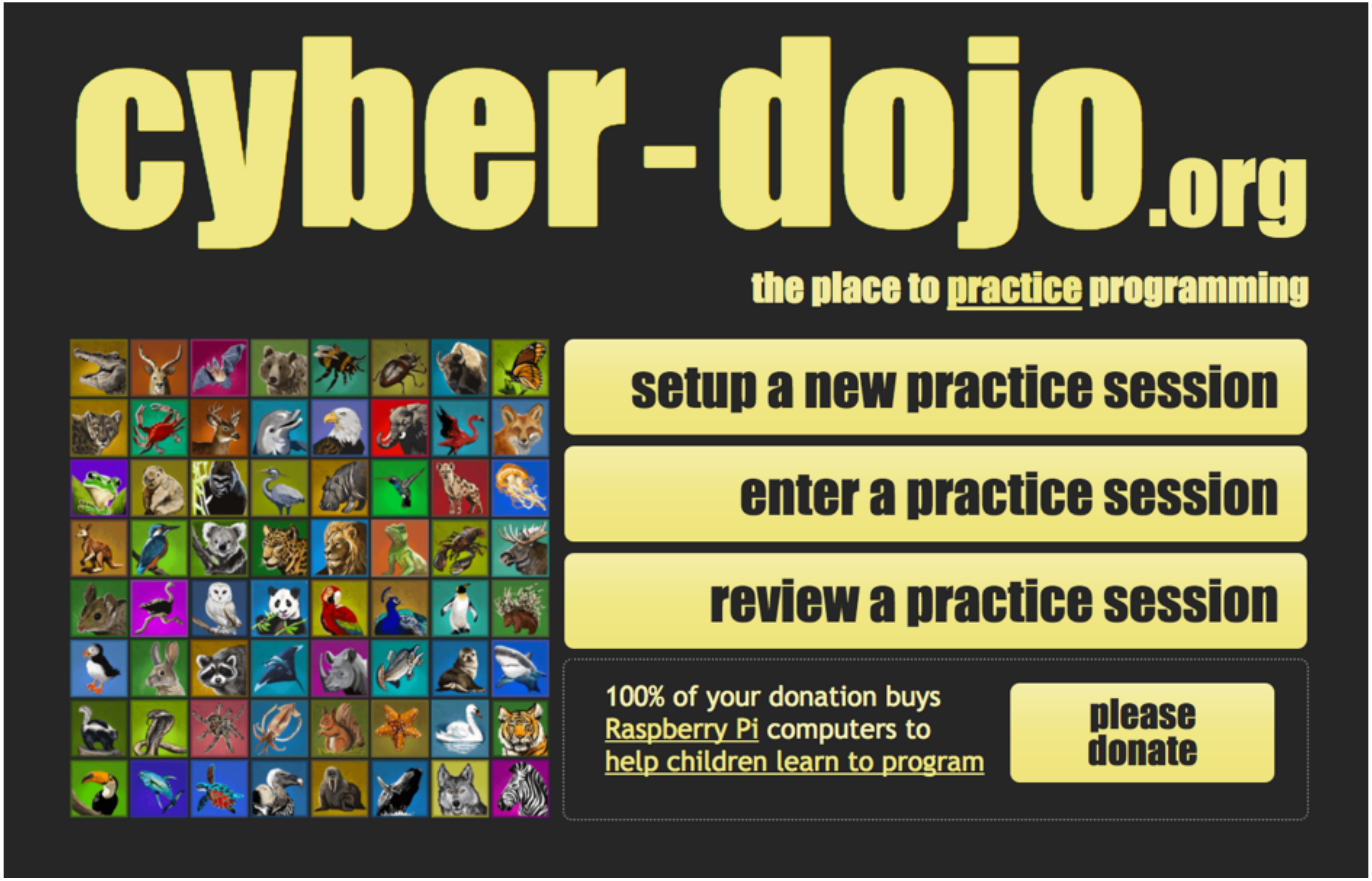
The $\overset{\text{ongoing}}{\vee}$ Design and Implementation of cyber-dojo

@JonJagger : twitter

cyber-dojo.org : practice

jon@jaggersoft.com : email

jonjagger.blogspot.co.uk : blog



cyber-dojo.org

the place to practice programming

setup a new practice session

enter a practice session

review a practice session

100% of your donation buys Raspberry Pi computers to help children learn to program

please donate

cyber-dojo Foundation

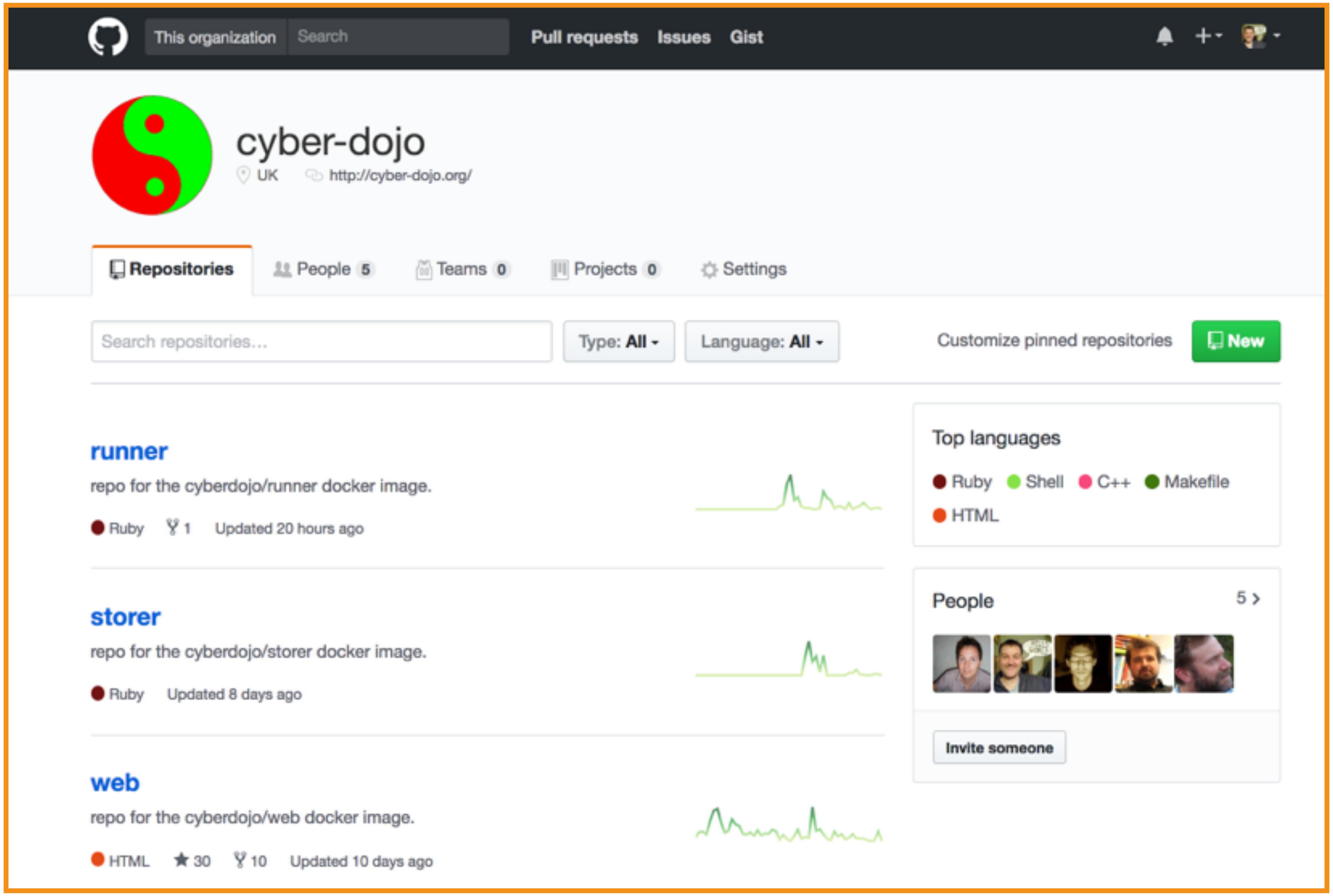
Commercial use of the public server requires a license
The cyberdojo Foundation issues licenses
100% of the license fees buy Raspberry Pi computers to help children learn to program
Hosting costs for the public server are paid by Cucumber Limited

Coimbatore, India

Bray, Ireland

Scottish Charitable Incorporated Organisation
(magic number SC045890)

open sourced



This organization Search Pull requests Issues Gist

cyber-dojo

UK http://cyber-dojo.org/

Repositories · People 5 · Teams 0 · Projects 0 · Settings

Search repositories... · Type: All · Language: All · Customize pinned repositories · New

runner

repo for the cyberdojo/runner docker image.

● Ruby · 1 · Updated 20 hours ago

storer

repo for the cyberdojo/storer docker image.

● Ruby · Updated 8 days ago

web

repo for the cyberdojo/web docker image.

● HTML · ★ 30 · 10 · Updated 10 days ago

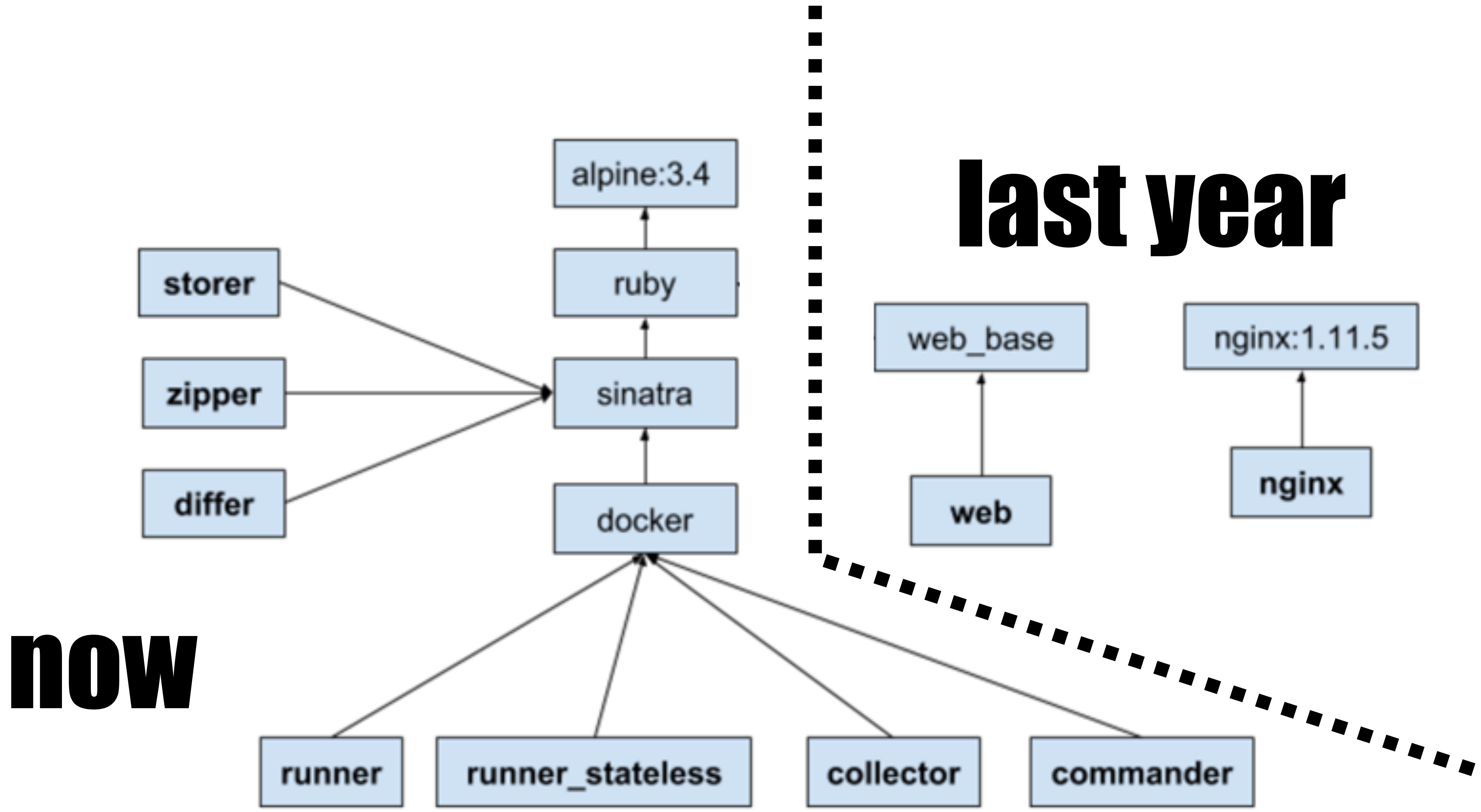
Top languages

● Ruby ● Shell ● C++ ● Makefile ● HTML

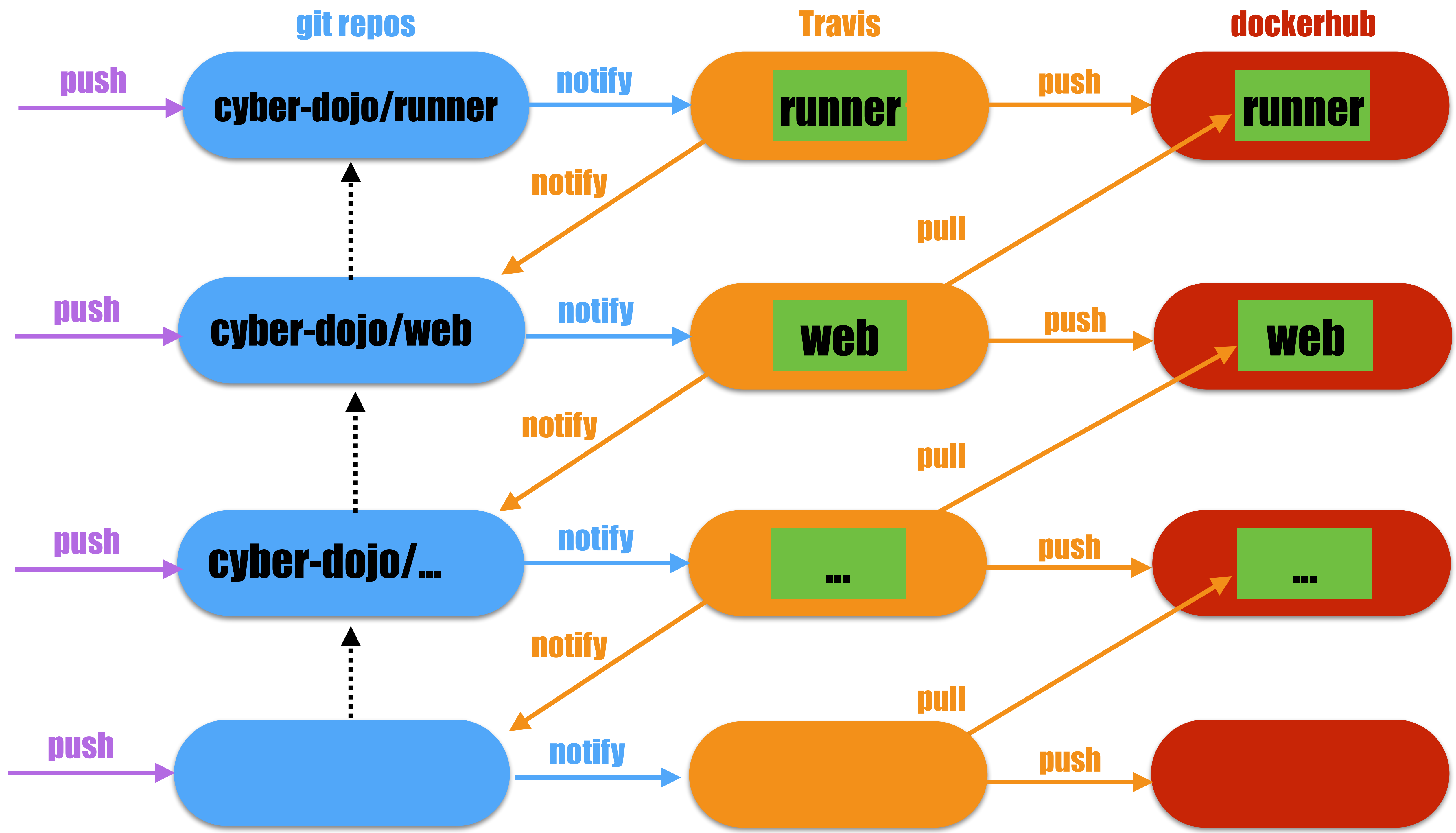
People 5 >

Invite someone

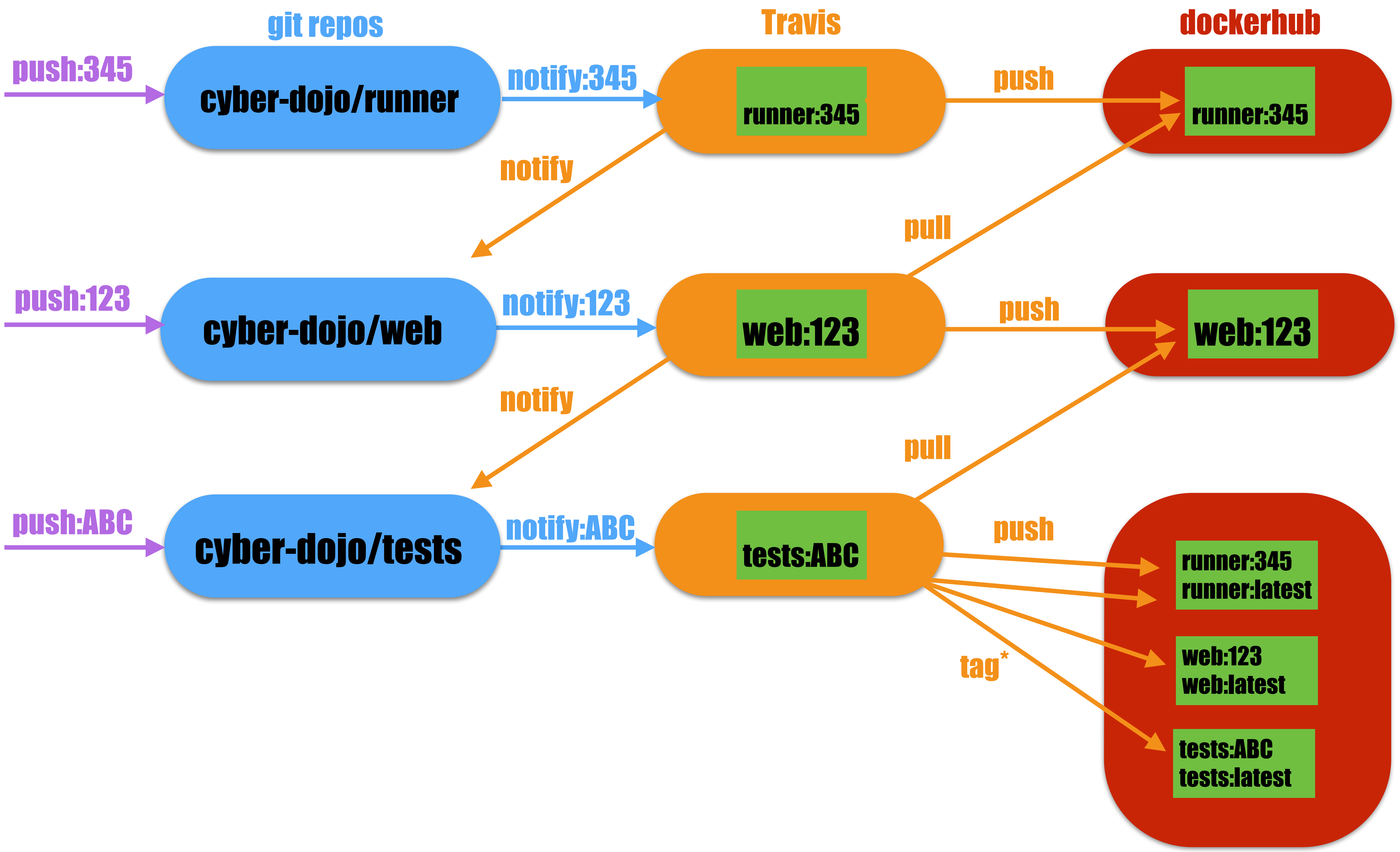
more decoupling



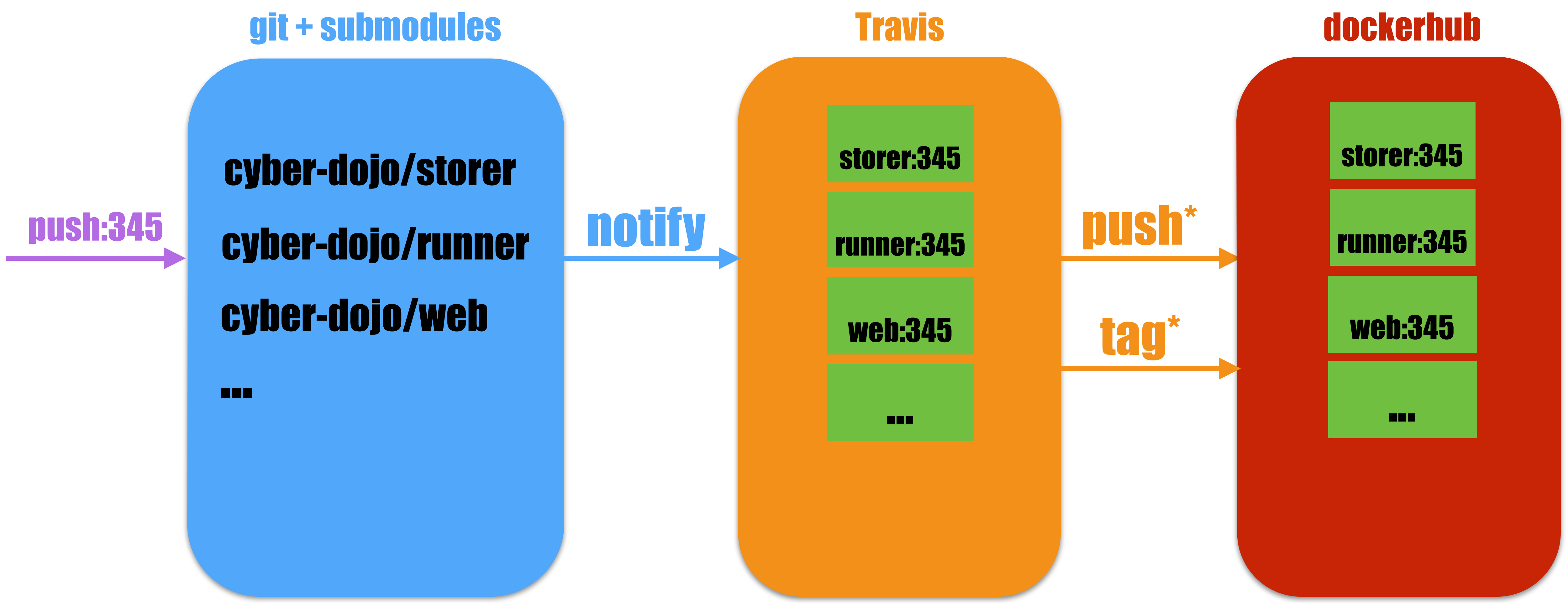
build architecture



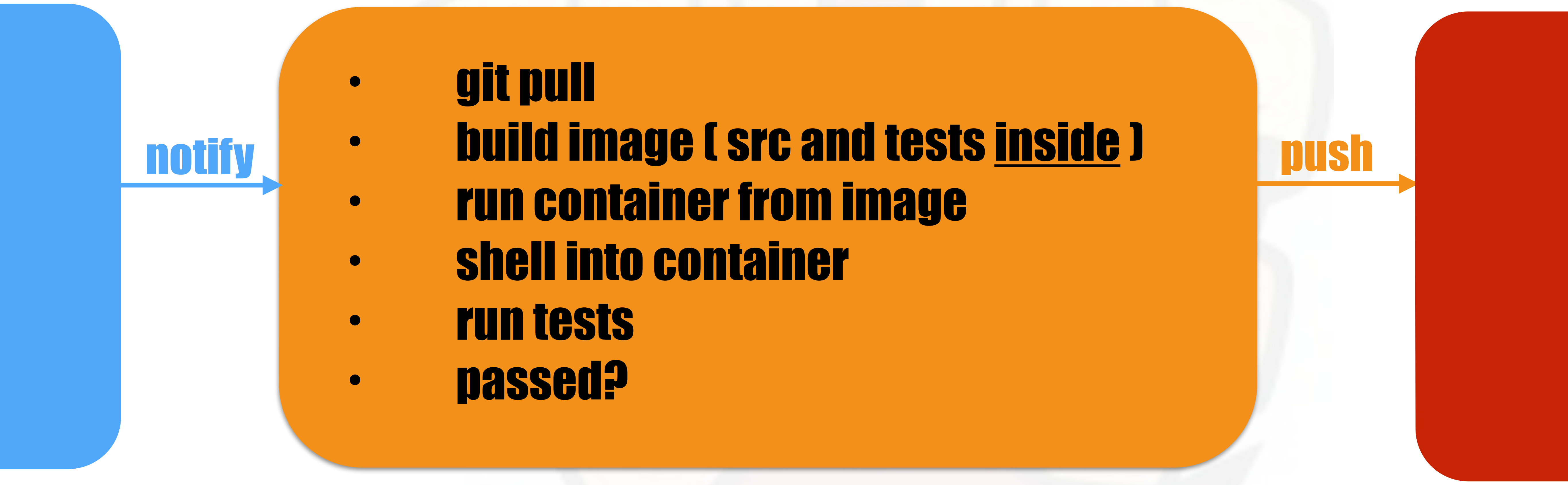
...build architecture



build architecture?

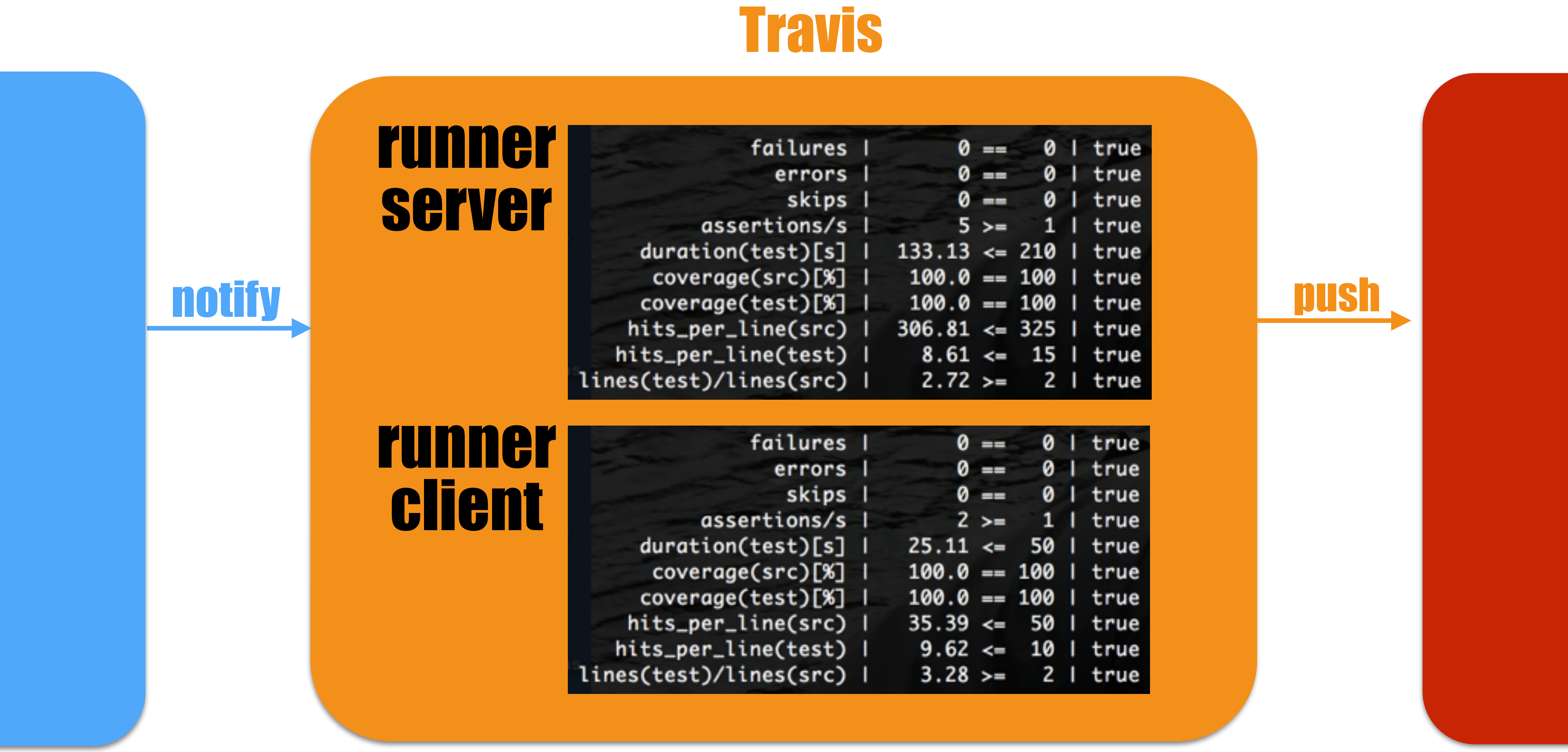


Travis



- git pull
- build image (src and tests inside)
- run container from image
- shell into container
- run tests
- passed?

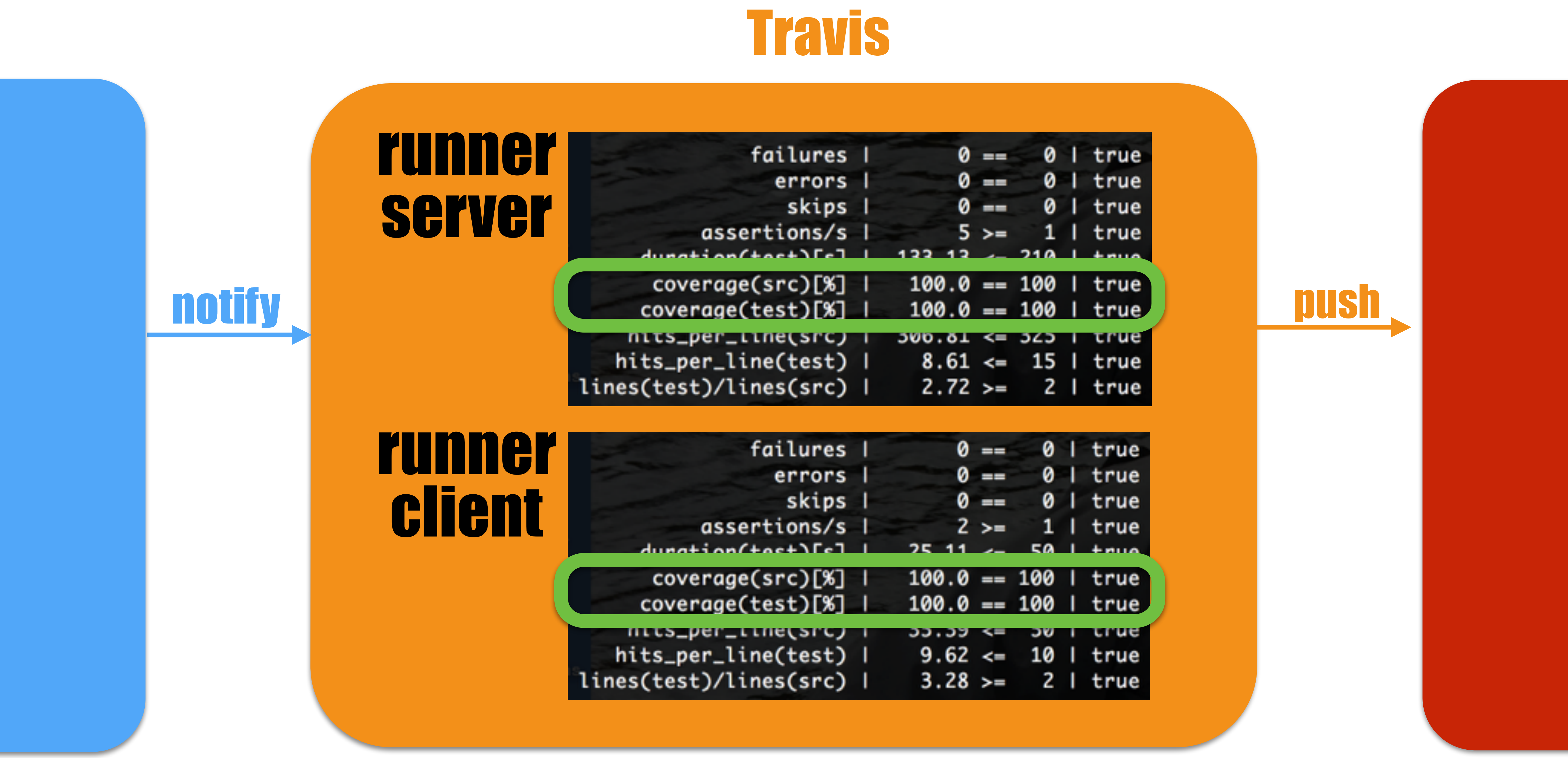
client-server testing



```
          failures |      0 ==   0 | true
            errors |      0 ==   0 | true
             skips |      0 ==   0 | true
      assertions/s |      5 >=   1 | true
  duration(test)[s] | 133.13 <= 210 | true
   coverage(src)[%] |  100.0 == 100 | true
  coverage(test)[%] |  100.0 == 100 | true
  hits_per_line(src) | 306.81 <= 325 | true
 hits_per_line(test) |   8.61 <=  15 | true
lines(test)/lines(src) |   2.72 >=   2 | true
```

```
          failures |      0 ==   0 | true
            errors |      0 ==   0 | true
             skips |      0 ==   0 | true
      assertions/s |      2 >=   1 | true
  duration(test)[s] |  25.11 <=  50 | true
   coverage(src)[%] |  100.0 == 100 | true
  coverage(test)[%] |  100.0 == 100 | true
  hits_per_line(src) |  35.39 <=  50 | true
 hits_per_line(test) |   9.62 <=  10 | true
lines(test)/lines(src) |   3.28 >=   2 | true
```

100% coverage



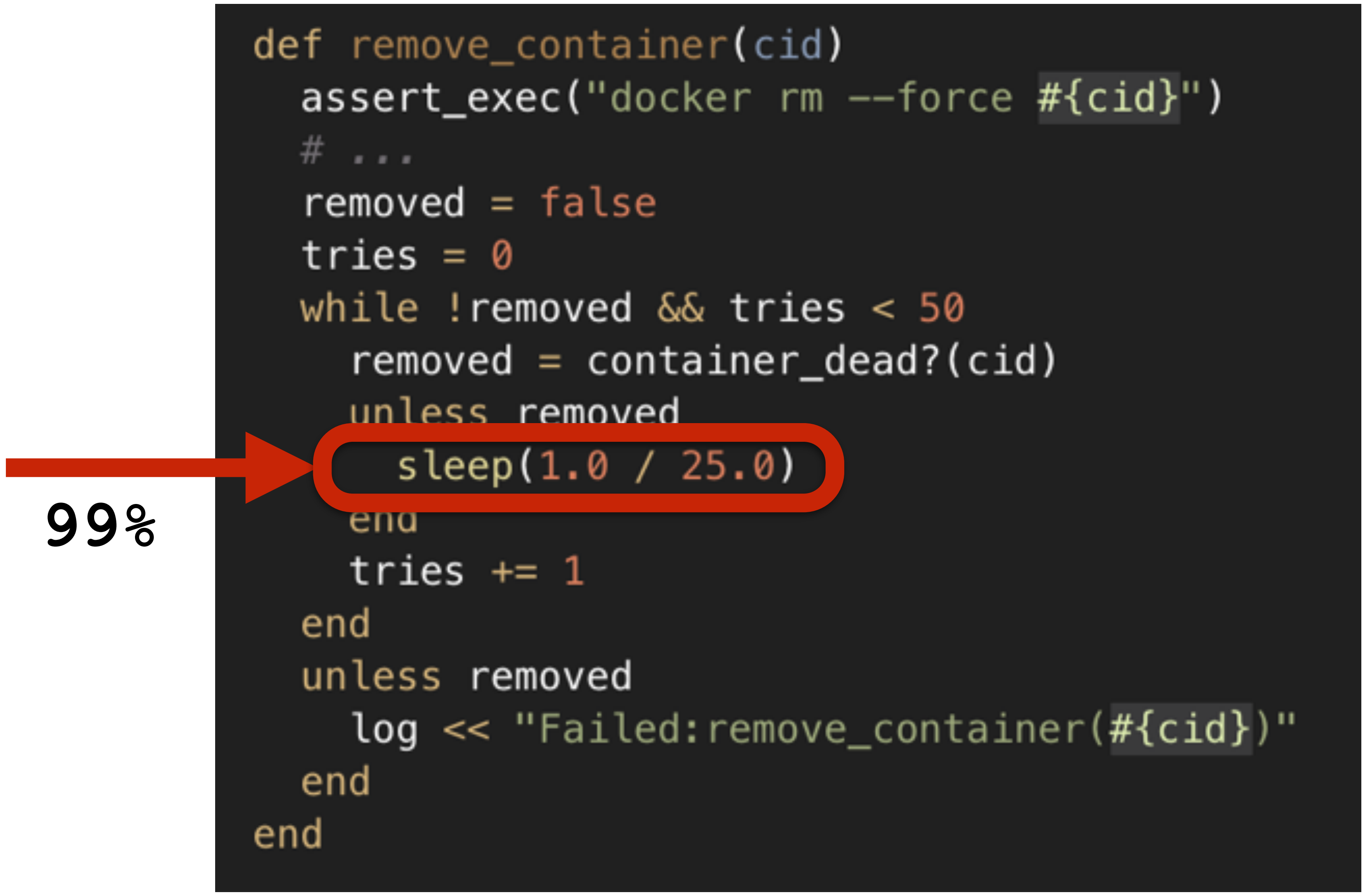
```
           failures |     0 ==    0 | true
             errors |     0 ==    0 | true
              skips |     0 ==    0 | true
       assertions/s |     5 >=    1 | true
  duration(test)[s] | 133.13 <=  210 | true
   coverage(src)[%] | 100.0 ==  100 | true
  coverage(test)[%] | 100.0 ==  100 | true
 hits_per_line(src) | 306.81 <= 325 | true
hits_per_line(test) |  8.61 <=   15 | true
lines(test)/lines(src) | 2.72 >=  2 | true
```

```
           failures |     0 ==    0 | true
             errors |     0 ==    0 | true
              skips |     0 ==    0 | true
       assertions/s |     2 >=    1 | true
  duration(test)[s] | 25.11 <=   50 | true
   coverage(src)[%] | 100.0 ==  100 | true
  coverage(test)[%] | 100.0 ==  100 | true
 hits_per_line(src) | 35.39 <=   50 | true
hits_per_line(test) |  9.62 <=   10 | true
lines(test)/lines(src) | 3.28 >=  2 | true
```

pro: refactoring

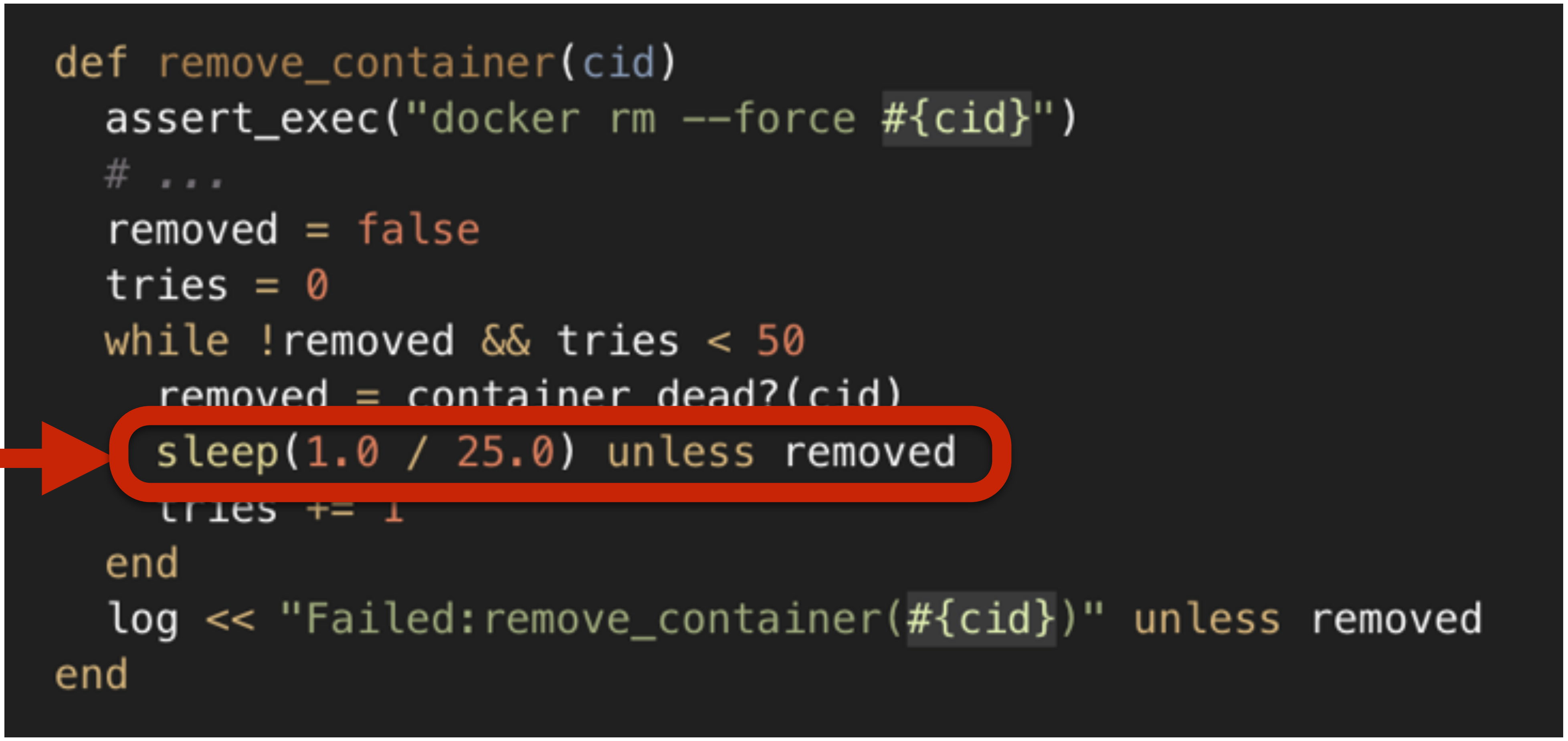
```
$ git log --grep='refactor'
    --format=oneline
    | wc
    | awk '{print $1}'

2263
```

pro: reveals poor design



```
def remove_container(cid)
  assert_exec("docker rm --force #{cid}")
  # ...
  removed = false
  tries = 0
  while !removed && tries < 50
    removed = container_dead?(cid)
    unless removed
      sleep(1.0 / 25.0)
    end
    tries += 1
  end
  unless removed
    log << "Failed:remove_container(#{cid})"
  end
end
```

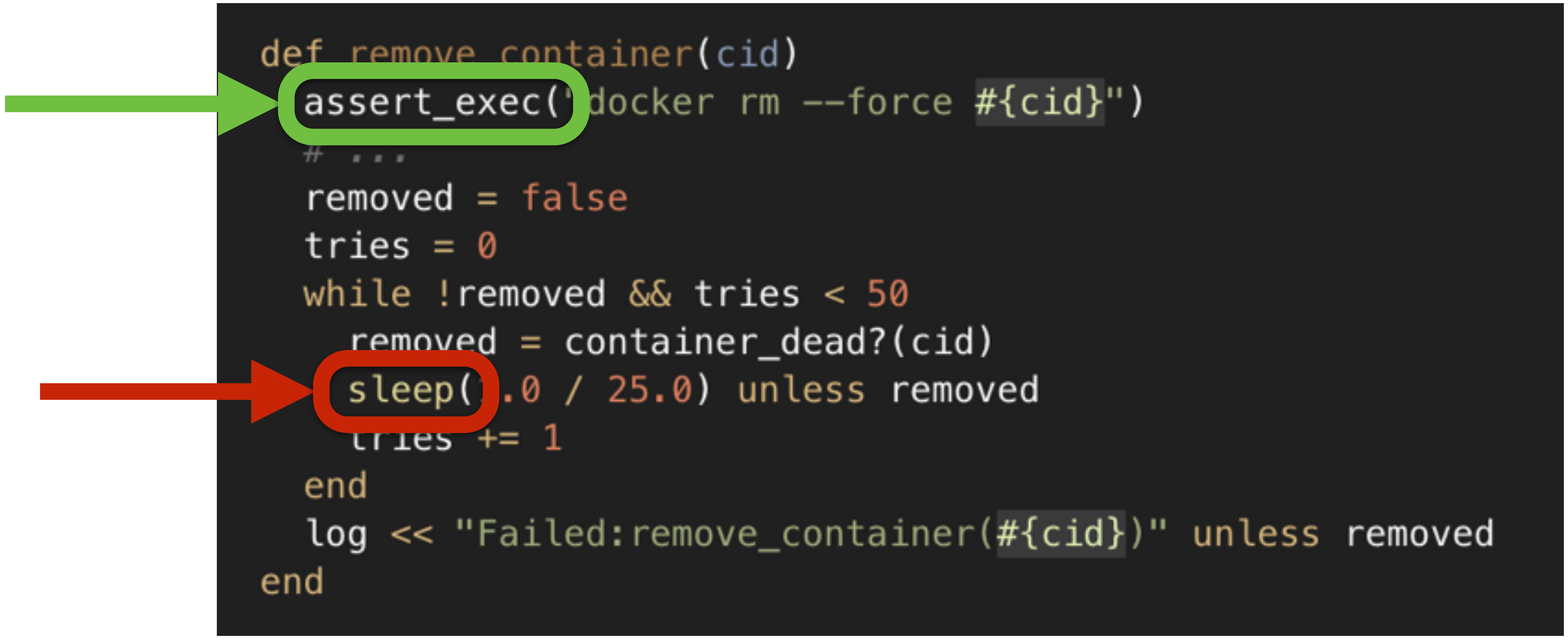
how not to fix it!



```
def remove_container(cid)
  assert_exec("docker rm --force #{cid}")
  # ...
  removed = false
  tries = 0
  while !removed && tries < 50
    removed = container_dead?(cid)
    sleep(1.0 / 25.0) unless removed
    tries += 1
  end
  log << "Failed:remove_container(#{cid})" unless removed
end
```

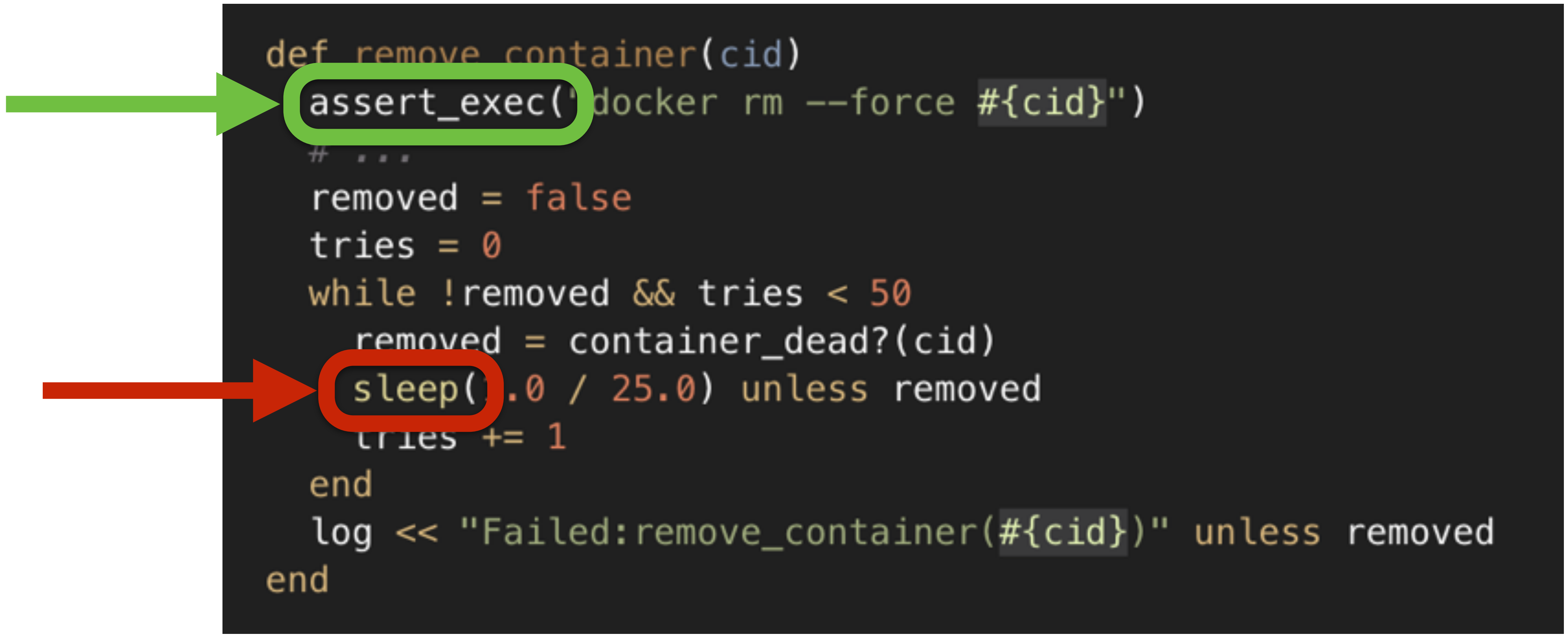
100%

Q: what does this tell us?



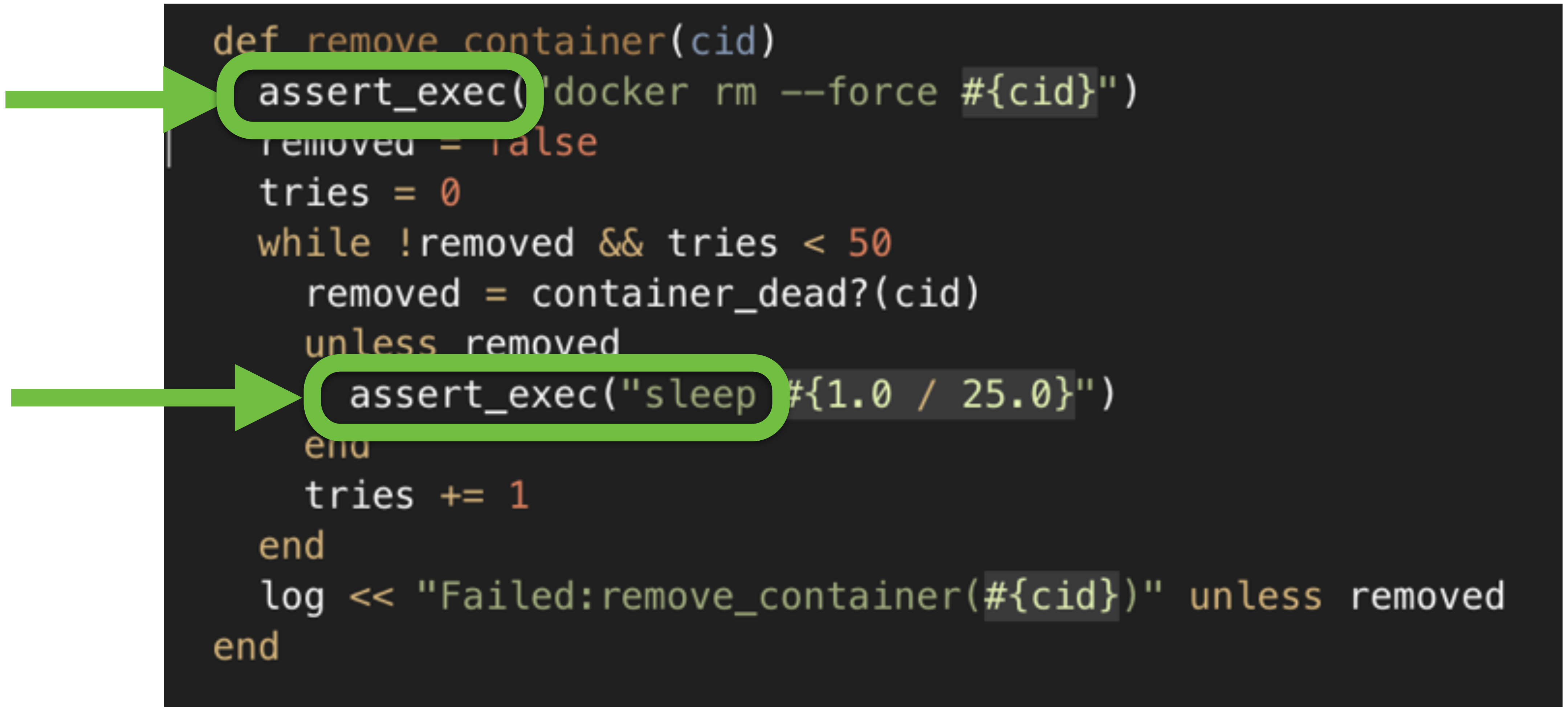
```ruby
def remove_container(cid)
  assert_exec("docker rm --force #{cid}")
  # ...
  removed = false
  tries = 0
  while !removed && tries < 50
    removed = container_dead?(cid)
    sleep(1.0 / 25.0) unless removed
    tries += 1
  end
  log << "Failed:remove_container(#{cid})" unless removed
end
```

A: different levels of abstraction



```
def remove_container(cid)
  assert_exec("docker rm --force #{cid}")
  # ...
  removed = false
  tries = 0
  while !removed && tries < 50
    removed = container_dead?(cid)
    sleep(1.0 / 25.0) unless removed
    tries += 1
  end
  log << "Failed:remove_container(#{cid})" unless removed
end
```

pro: design pressure

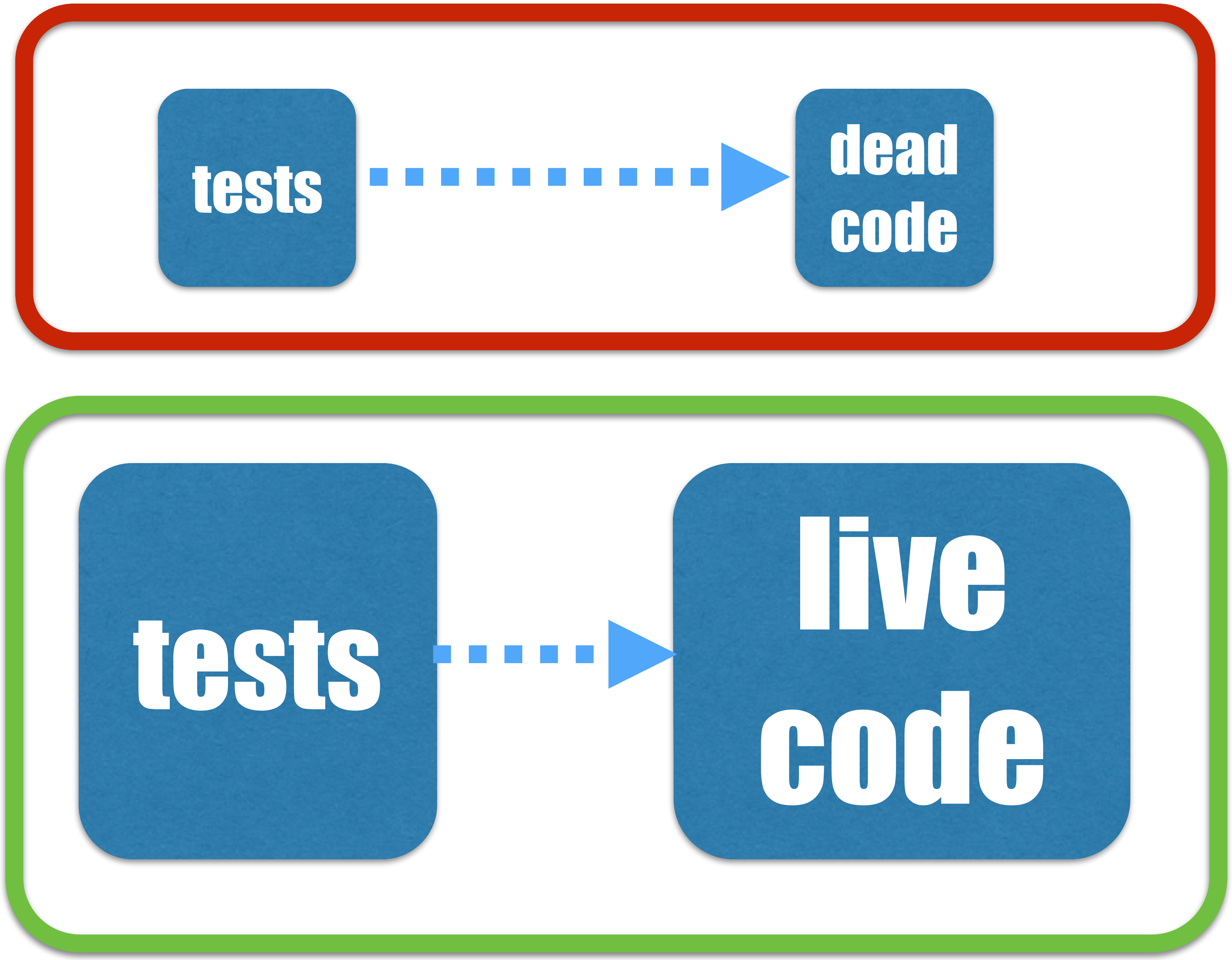


```ruby
def remove_container(cid)
  assert_exec("docker rm --force #{cid}")
  removed = false
  tries = 0
  while !removed && tries < 50
    removed = container_dead?(cid)
    unless removed
      assert_exec("sleep #{1.0 / 25.0}")
    end
    tries += 1
  end
  log << "Failed:remove_container(#{cid})" unless removed
end
```

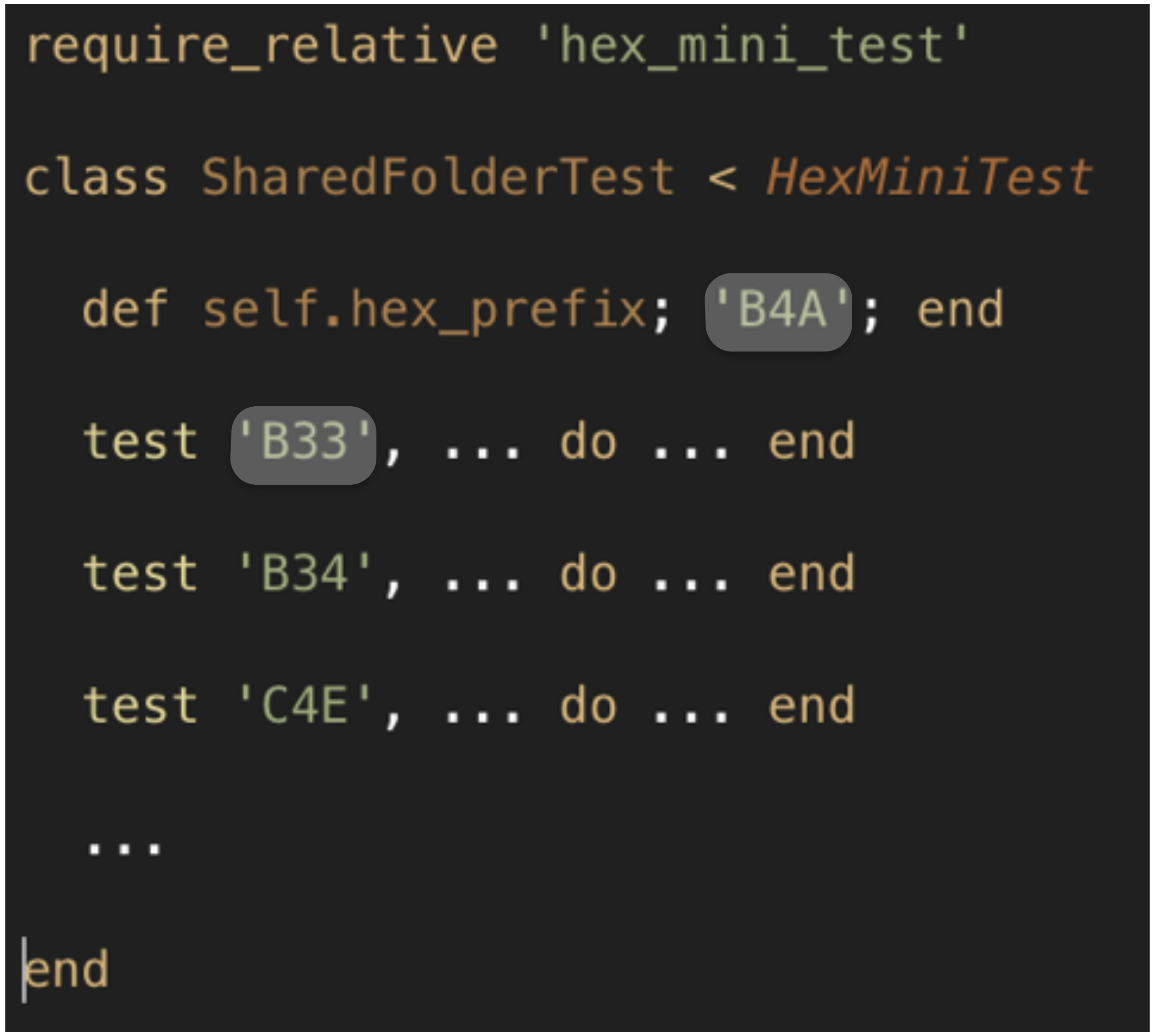
pro: deleting dead code

```
$ git log --grep='delete'
    --format=oneline
    | wc
    | awk '{print $1}'

353
```

con: zombie code



custom ruby testing-framework



```ruby
require_relative 'hex_mini_test'

class SharedFolderTest < HexMiniTest

  def self.hex_prefix; 'B4A'; end

  test 'B33', ... do ... end

  test 'B34', ... do ... end

  test 'C4E', ... do ... end

  ...

end
```

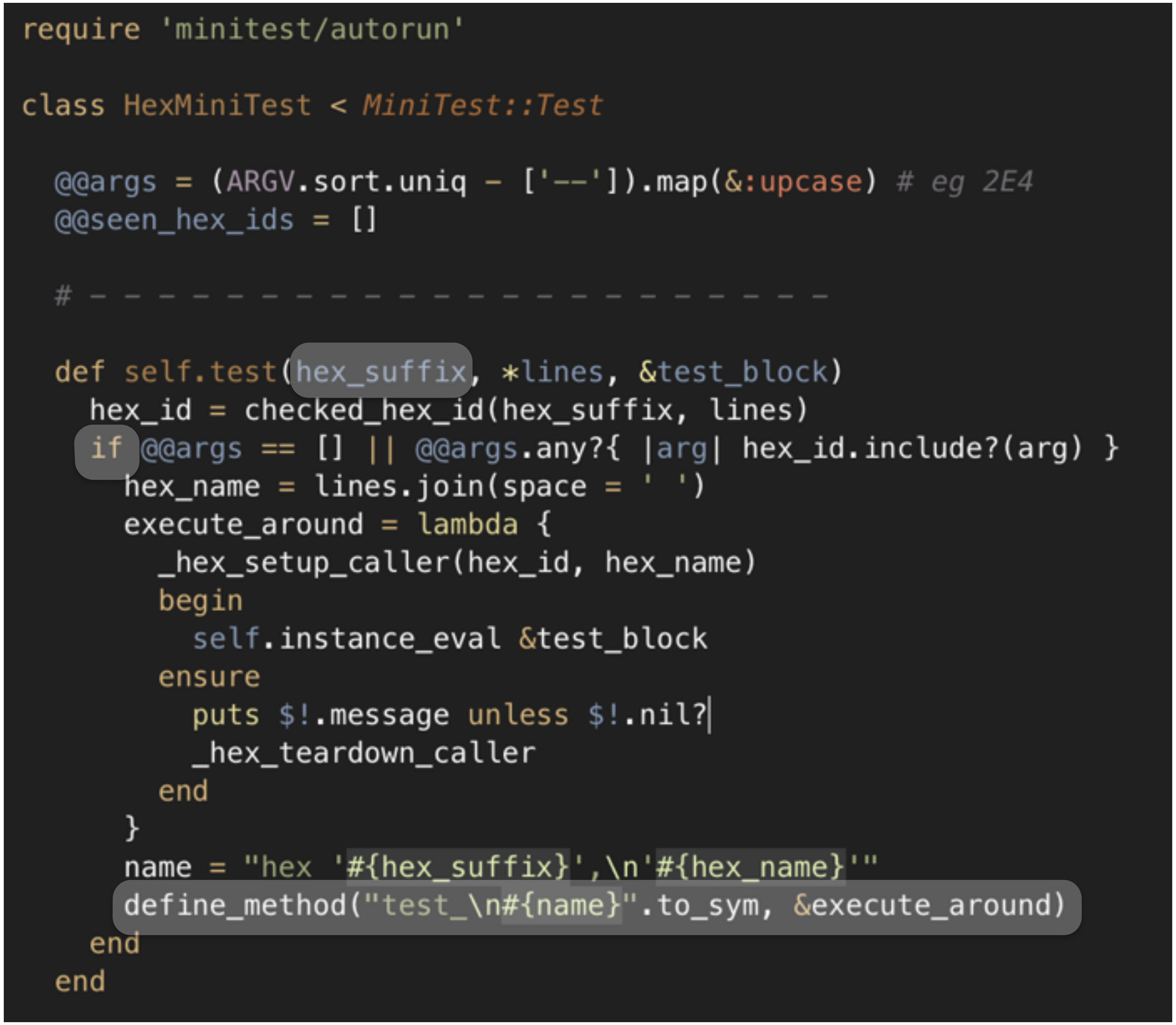
```
$ ./pipe_build_up_test.sh
 645 assertions...    59.8s
```

```
$ ./pipe_build_up_test.sh B4A
 12 assertions...     7.3s
```

```
$ ./pipe_build_up_test.sh B3
  5 assertions...     2.1s
```

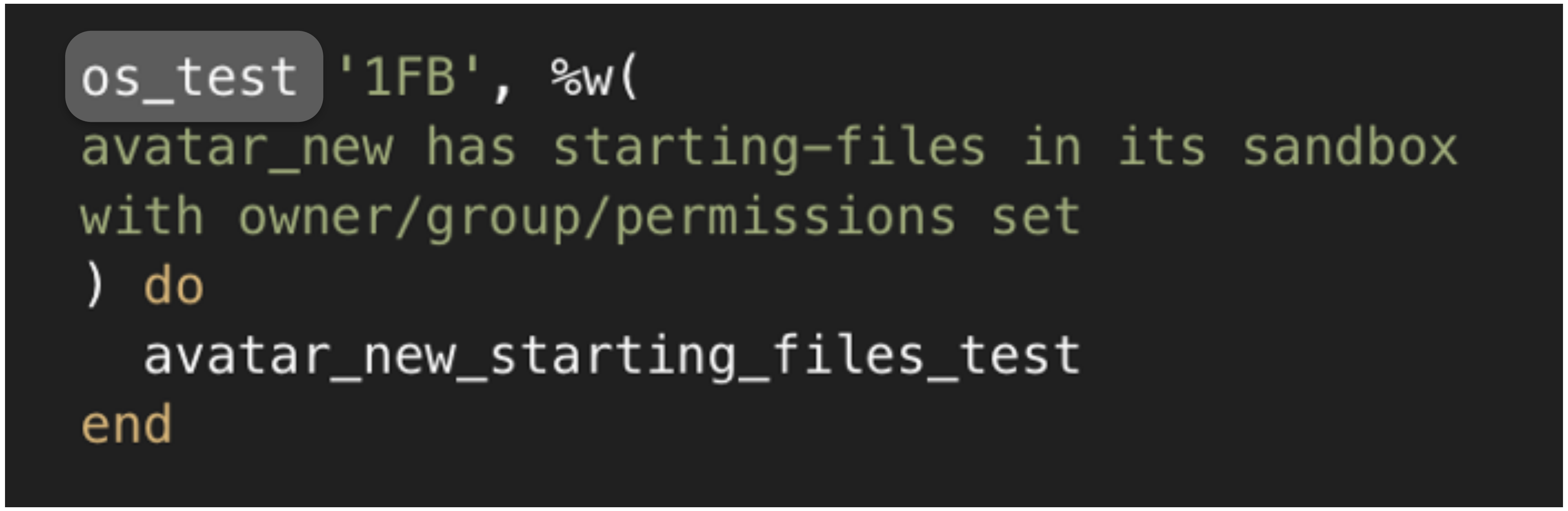
```
$ ./pipe build up test.sh B33
  3 assertions...     1.6s
```

custom ruby testing-framework

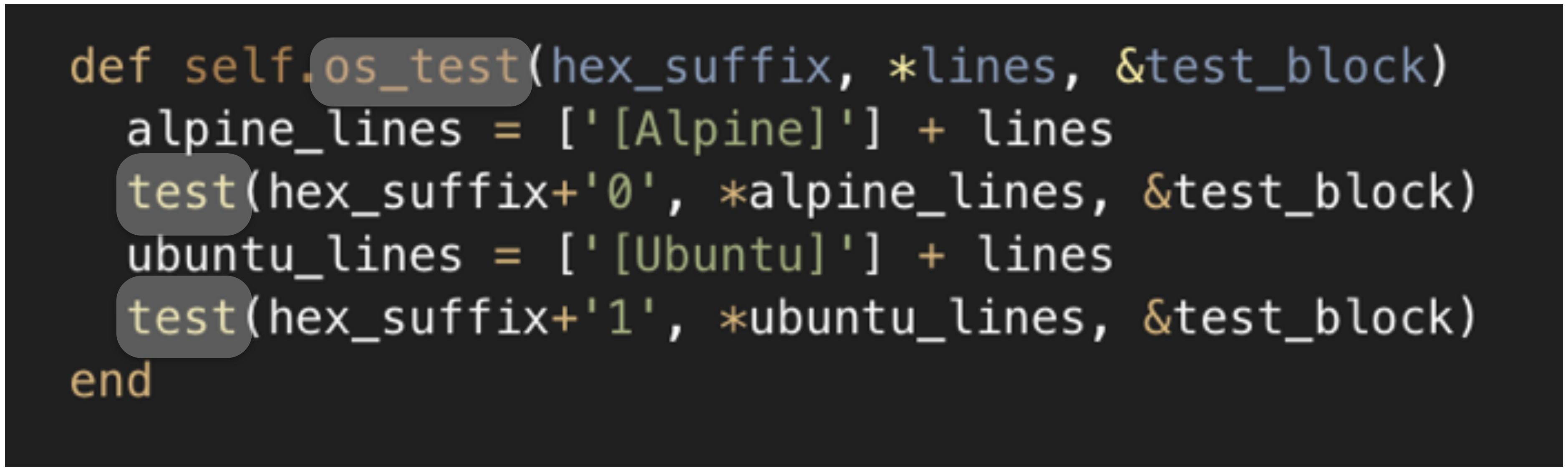


```ruby
require 'minitest/autorun'

class HexMiniTest < MiniTest::Test

  @@args = (ARGV.sort.uniq - ['--']).map(&:upcase) # eg 2E4
  @@seen_hex_ids = []

  # - - - - - - - - - - - - - - - - - - - - - -

  def self.test(hex_suffix, *lines, &test_block)
    hex_id = checked_hex_id(hex_suffix, lines)
    if @@args == [] || @@args.any?{ |arg| hex_id.include?(arg) }
      hex_name = lines.join(space = ' ')
      execute_around = lambda {
        _hex_setup_caller(hex_id, hex_name)
        begin
          self.instance_eval &test_block
        ensure
          puts $!.message unless $!.nil?
          _hex_teardown_caller
        end
      }
      name = "hex '#{hex_suffix}',\n'#{hex_name}'"
      define_method("test_\n#{name}".to_sym, &execute_around)
    end
  end
```

custom ruby testing-framework

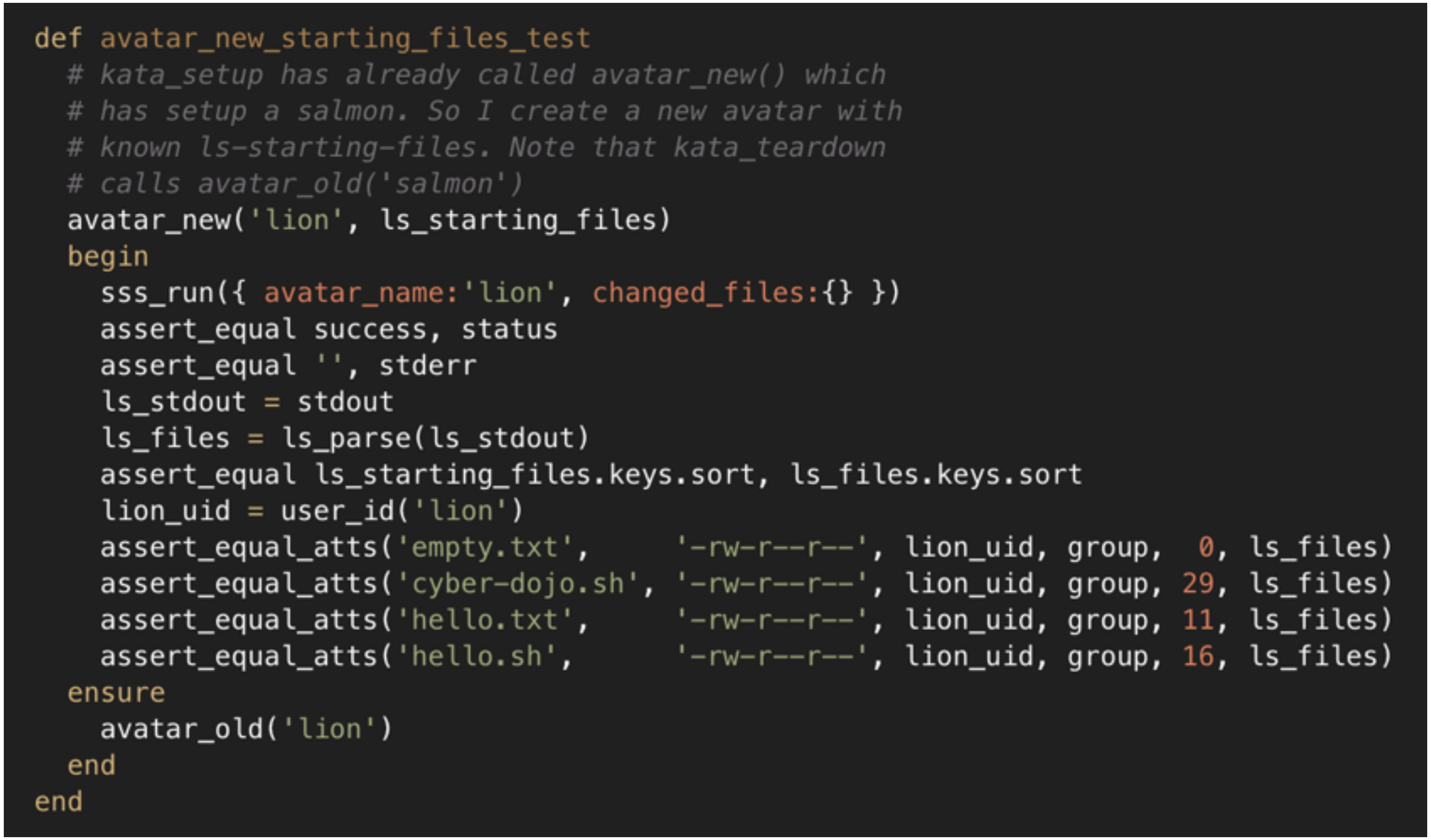


```ruby
os_test '1FB', %w(
avatar_new has starting-files in its sandbox
with owner/group/permissions set
) do
  avatar_new_starting_files_test
end
```



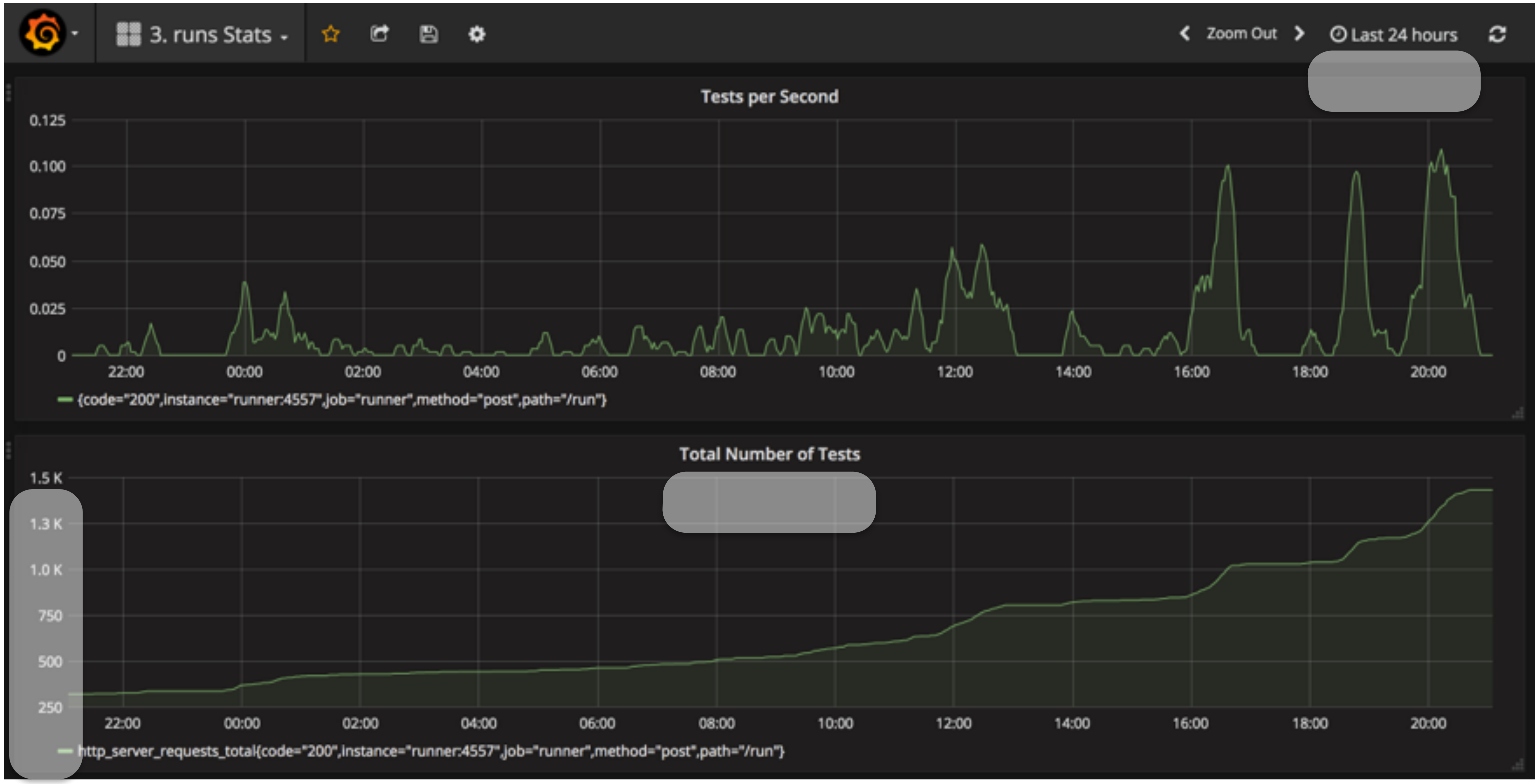
```ruby
def self.os_test(hex_suffix, *lines, &test_block)
  alpine_lines = ['[Alpine]'] + lines
  test(hex_suffix+'0', *alpine_lines, &test_block)
  ubuntu_lines = ['[Ubuntu]'] + lines
  test(hex_suffix+'1', *ubuntu_lines, &test_block)
end
```

custom ruby testing-framework



```ruby
def avatar_new_starting_files_test
  # kata_setup has already called avatar_new() which
  # has setup a salmon. So I create a new avatar with
  # known ls-starting-files. Note that kata_teardown
  # calls avatar_old('salmon')
  avatar_new('lion', ls_starting_files)
  begin
    sss_run({ avatar_name:'lion', changed_files:{} })
    assert_equal success, status
    assert_equal '', stderr
    ls_stdout = stdout
    ls_files = ls_parse(ls_stdout)
    assert_equal ls_starting_files.keys.sort, ls_files.keys.sort
    lion_uid = user_id('lion')
    assert_equal_atts('empty.txt',     '-rw-r--r--', lion_uid, group,  0, ls_files)
    assert_equal_atts('cyber-dojo.sh', '-rw-r--r--', lion_uid, group, 29, ls_files)
    assert_equal_atts('hello.txt',     '-rw-r--r--', lion_uid, group, 11, ls_files)
    assert_equal_atts('hello.sh',      '-rw-r--r--', lion_uid, group, 16, ls_files)
  ensure
    avatar_old('lion')
  end
end
```

monitoring



{code="200",instance="runner:4557",job="runner",method="post",path="/run"}

http_server_requests_total{code="200",instance="runner:4557",job="runner",method="post",path="/run"}

@JonJagger : twitter

cyber-dojo.org : practice

jon@jaggersoft.com : email

jonjagger.blogspot.co.uk : blog