Time brings all things to pass

Kamil Witecki
Idle time is wasted time

Before

After
Where it all started

Memory (MiB)
What we will need?

Time

9 10 11 12 1 2 3 4

Model

Capacity

0,00% 20,00% 40,00% 60,00% 80,00% 100,00% 120,00% 250 750 1250 1750 2250 2750 3250 3750 4250 4750 5250 5750 6250 6750 7250 7750 8250 8750 9250 9750
What we will need?

Time

Model
What we will need?

Time

Model

CPU (%)

Capacity

0,00% 20,00% 40,00% 60,00% 80,00% 100,00% 120,00%
Schedule is CPU limited

Before

After
TEST(EventsSentAroundTimeout)
{
    //force ping before timeout???
    //force ping after timeout???
}
```javascript
local state = ""
timer.setTimeout(100, function ()
  state = "ping"
timer.setTimeout(100, function ()
  state = "pong"
end) end)
timer.setTimeout(1000, function ()
  if state ~= "pong" then
    print "FAILED"
    os.exit(1)
  end
  print "OK"
end)
```
patch.lua:

```lua
function(f) return function(t, cb)
    local n = t/1000
    f(n, cb)
end
end)(timer.setTimeout)
```

Easy, broken attempt
> luvit code.lua
OK

> luvit -e "local x = require './patch.lua'" correct.lua
FAILED
What happens?

Time

Timeouts

Events

State

Time

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
Normal execution

Timeouts

Events

Main

Time

State

\[0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ 11\ 12\ 13\ 14\ 15\ 16\ 17\ 18\ 19\ 20\ 21\ 22\]
What happens?

Timeouts

Events

Time

State
Shortened timeouts

Timeouts

Events

Main

Time

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

State
Observations?

- Speeding up time does not speed up how fast work is done.
- Time resolution is finite.
DAG of events

1000ms

100ms

Main

Ping

+100ms

Pong

1000ms

Verify
DAG of events

1000ms

Main

100ms

Ping

100ms

+100ms

Verify

0 1 2 3 4 ...

10 11
DAG of events

- Main
  - 1000ms
  - 100ms
  - +100ms

- Ping
  - 100ms
  - +100ms

- Pong
  - +100ms
DAG of events

- **Main**
  - **Ping**: 1000ms
  - **Pong**: 100ms
  - **Verify**: 1000ms

  - **100ms**
  - **+100ms**

- **Main**: 100ms
What happens when we shorten time
Reliable code must expand the window
Forget about wall clock!
Disconnect from wall clock

patch.lua:

```
local timeouts = pqueue.new(function (a, b)
    return a[1] < b[1]
end
, {})

timer.setTimeout = (function ()
    return function (t, cb)
        local timeout = {t+reference_time, cb, 0}
        timeouts:push(timeout)
    end
end)()
```
If not wall clock, then what?

Main

1000ms

100ms

100ms

Ping

+100ms

100ms

Pong

+100ms

1000ms

Verify

1000ms
If not wall clock, then what?

1000ms

Main

100ms

100ms

Ping

+100ms

Pong

+100ms

Verify

1000ms

Events
But how to do that?

- Where from to check the queue?
  - From separate thread,
  - From queue itself,

- How to tell if queue is empty or not?
  - There is explicit empty() or size(),
  - We need to drain it,
But how to do that?

- Where from to check the queue?
  - From separate thread,
  - From queue itself,

- How to tell if queue is empty or not?
  - There is explicit empty() or size(),
  - We need to drain it,
Checking queue state externally...

```cpp
std::thread forwarder {
    [](event_queue &equeue, timeout_queue &tqueue) {
        while (!equeue.empty() || !tqueue.empty()) {
            if (equeue.empty()) {
                equeue.push(tqueue.front());
                tqueue.pop();
            }
            std::this_thread::sleep_for(50ms);
        }, equeue, tqueue };
```
... might not end well

```cpp
std::thread processor {
    [](event_queue &equeue, timeout_queue &tqueue) {
        while (!equeue.empty() || !tqueue.empty()) {
            if (!equeue.empty()) {
                auto event = equeue.front();
                equeue.pop();
                dispatch(event);
            }
            else std::this_thread::sleep_for(50ms);
        }
    }, equeue, tqueue
};
```
event_type guard = [&guard, &equeue, &tqueue] {
    if (equeue.empty()) {
        if (!tqueue.empty()) {
            equeue.push(tqueue.front());
            tqueue.pop();
        }
    } else equeue.push(guard);
};
equeue.push(guard);
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?
What if we cannot check if it is empty?

Events
What if we cannot check if it is empty?
What if we cannot check if it is empty?
```cpp
int debounce = 500;

event_type guard =
    [debounce, &guard, &equeue, &tqueue]() mutable {
        if (--debounce == 0) {
            if (!tqueue.empty()) {
                equeue.push(tqueue.front());
                tqueue.pop();
            }
        } else equeue.push(guard);
    };

equeue.push(guard);
```
Let’s see the guard at work!
Let's see the guard at work!
Let’s see the guard at work!
Let's see the guard at work!
Let's see the guard at work!
Let’s see the guard at work!
Let’s see the guard at work!
Let’s see the guard at work!
Let’s see the guard at work!
Let's see the guard at work!

Events
Let's see the guard at work!

Events
Let’s see the guard at work!
Let's see the guard at work!
Let’s see the guard at work!
local state = ""
timer.setTimeout(100, function ()
    state = "ping"
timer.setTimeout(100, function ()
    state = "pong"
end) end)
timer.setTimeout(1000, function ()
    if state ~= "pong" then
        print "FAILED"
os.exit(1)
    end
    print "OK"
end)
And naive attempt at solving the problem?

patch.lua:

```lua
local n = t/1000
f(n, cb)

end

end)(timer.setTimeout)
```
This is him now.

```javascript
function module.ensureEmptyQueue(n, f) {
    if (n > 0) {
        timer.setImmediate(function () {
            module.ensureEmptyQueue(n - 1, f);
        });
    } else {
        f();
    }
}

module.ensureEmptyQueue(500, function () {
    if (#timeouts > 0) {
        timeouts:pop();
    }
});
```
Results?

```lua
> luvit code.lua
OK
real 0m1.036s

> luvit -e "local x= require './patch.lua'" correct.lua
OK
real 0m0.109s

> luvit -e "print 'x'"
x
real 0m0.108s
```
What we did
What we did
To turn idle time into productive time

Before

After
Questions and Answers

Kamil.Witecki@nokia.com
frankieleon. Say what?
https://www.flickr.com/photos/armydre2008/19339086155, 2015. See licence: https://creativecommons.org/licenses/by/2.0/ (Attribution 2.0 Generic (CC BY 2.0)).