



Scheduled jobs invocation emulation for test environments

Sidekiq::Portal

https://github.com/0exp/sidekiq_portal

Sidekiq::Portal? Сначала кто ты :)



 Coding Everywhere

Rustam Ibragimov

0exp

Edit profile

Software Architecture Lord, Ruby Wizard, OpenSource Adept. @smart-rb

 @umbrellio @smart-rb

 Russia, Saint Petersburg

 <https://github.com/0exp>

 **qonfig**

Config. Defined as a class. Used as an instance. Lazy instantiation. Validation layer. Thread-safe. Support for YAML, TOML, JSON, __END__, ENV. Extremely simple to define. Extremely simple to use.

 Ruby ★ 12 🍴 2

 **any_cache**

A simplest cache wrapper that provides a minimalistic generic interface for all well-known cache storages. You can use any cache implementation in ANY project easily.


 Ruby ★ 1

 **smart_core**

Powerful set of common abstractions: Service Object (Operation), IoC Container (Dependency Container), Validation Object, Initialization DSL (and more..) (in active development)

 Ruby ★ 10 🍴 3

 **sidekiq_portal**

 Scheduled jobs invocation emulation for test environments (eliminate time traveling by might and magic 🤖)

 Ruby ★ 3 🍴 2

 **symbiont-ruby**

Invoke proc-objects in many contexts simultaneously. Provides a controllable technique to intercept and dispatch methods inside proc object (or inside a series of proc objects)

 Ruby ★ 6

 **siege**

Siege - software architecture principles realized as a code (in active development)

 Ruby ★ 1

https://github.com/0exp/sidekiq_portal

Sidekiq::Portal - Какую проблему решает?

Sidekiq::Portal - Какую проблему решает?

- у нас есть **Sidekiq::Sheduler** (а еще **Sidekiq::Cron**)

Sidekiq::Portal - Какую проблему решает?

- у нас есть **Sidekiq::Sheduler** (а еще **Sidekiq::Cron**)
 - мы обожаем писать тесты и делаем это хорошо (да?)

Sidekiq::Portal - Какую проблему решает?

- у нас есть **Sidekiq::Sheduler** (а еще **Sidekiq::Cron**)
- мы обожаем писать тесты и делаем это хорошо (да?)
- мы любим **Timecop**, особенно **Timecop.travel** и **Timecop.freeze**

Sidekiq::Portal - Какую проблему решает?

- у нас есть **Sidekiq::Sheduler** (а еще **Sidekiq::Cron**)
 - мы обожаем писать тесты и делаем это хорошо (да?)
 - мы любим **Timecop**, особенно **Timecop.travel** и **Timecop.freeze**
 - но Timecop не любит scheduler'ы 😞 - джобки не запускаются

Sidekiq::Portal - Какую проблему решает?

- у нас есть **Sidekiq::Sheduler** (а еще **Sidekiq::Cron**)
 - мы обожаем писать тесты и делаем это хорошо (да?)
 - мы любим **Timecop**, особенно **Timecop.travel** и **Timecop.freeze**
 - но Timecop не любит scheduler'ы 😞 - джобки не запускаются
 - на рынке Github нет решений, а RubyGems - молчит



Так а в чем проблема?

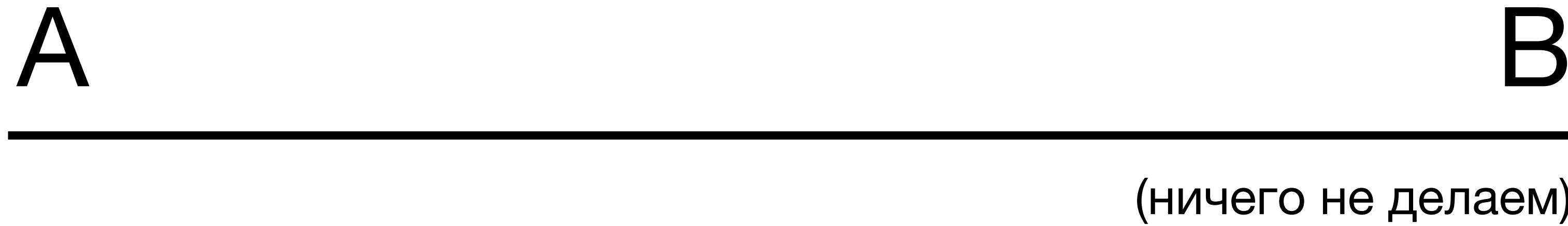
```
-BalanceWithdrawJob ({ every: "15m" })  
-Timecop.travel(Time.current + 2.hours)
```

A

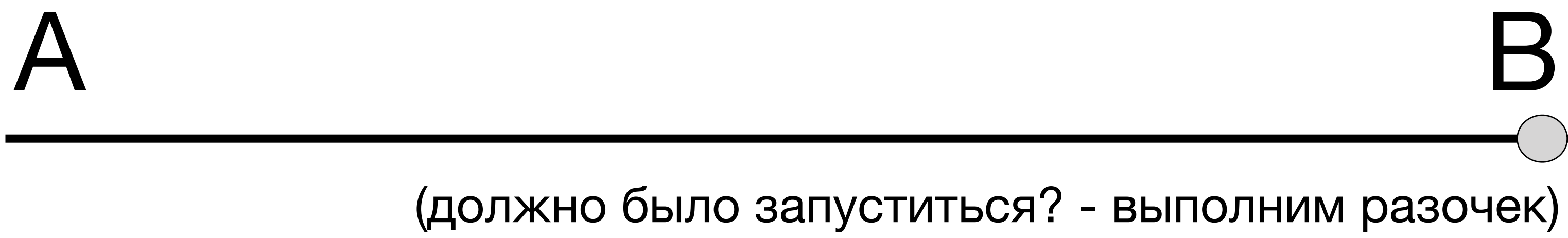
B

(ничего не делаем)

`-BalanceWithdrawJob ({ every: "15m" })`
`-Timecop.travel(Time.current + 2.hours)`




`-BalanceWithdrawJob ({ every: "15m" })`
`-Timecop.travel(Time.current + 2.hours)`







Заметили проблему?



**Time . current
(Time . now)**



- BalanceWithdrawJob ({ every: "15m" })
- Timecop.travel(Time.current + 2.hours)

A

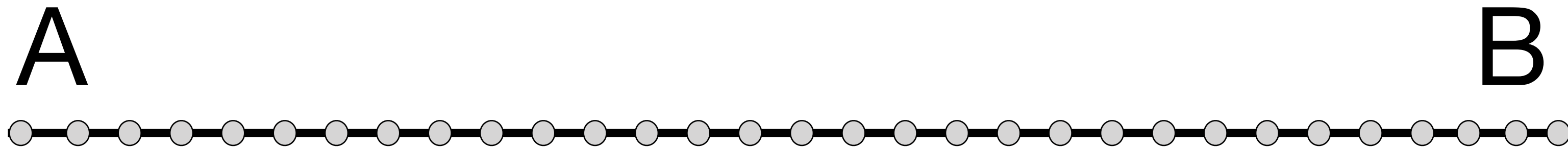
B

1) перехватываем **Timecop.travel**

```
-BalanceWithdrawJob ({ every: "15m" })  
-Timecop.travel(Time.current + 2.hours)
```



1) перехватываем **Timecop.travel**

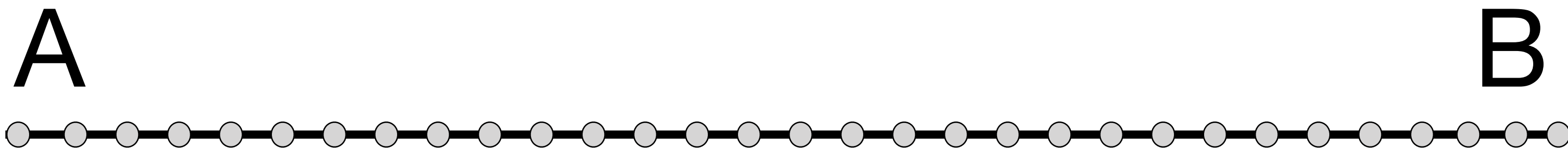


2) строим **таймлайн**, согласно планировщику (**таймпоинты**)

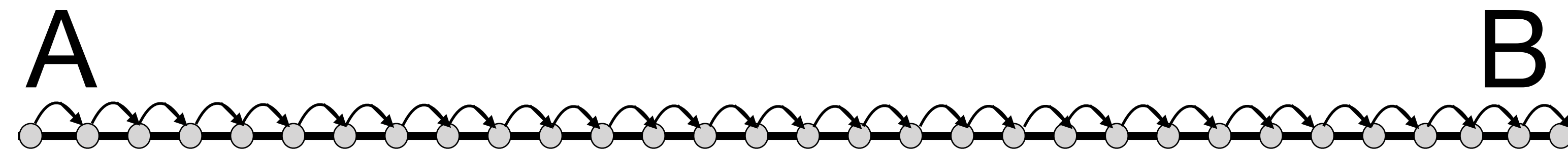
```
-BalanceWithdrawJob ({ every: "15m" })  
-Timecop.travel(Time.current + 2.hours)
```



1) перехватываем **Timecop.travel**



2) строим **таймлайн**, согласно планировщику (**таймпоинты**)



3) прыгаем по таймпоинтам и выполняем все джобки в очереди

```
Timecop.freeze(time_point) { run_job }
```

Пример

Пример (ставим гемчик)

```
gem 'sidekiq', '>= 5' # runtime dependency
gem 'timecop', '~> 0.9' # runtime dependency

group :test do
  gem 'rspec'
  gem 'sidekiq-portal'
end
```

```
bundle install
```

```
require 'timecop' # runtime dependency
require 'sidekiq' # runtime dependency
require 'sidekiq/api'
require 'sidekiq/testing'

require 'sidekiq_portal'
```

Пример (наша джобка)

```
class HookExampleJob < ApplicationJob
  def perform
    GLOBAL_HOOK_INTERCEPTOR << Time.current # intercept current time
  end
end
```

```
:schedule:
  HookExample:
    every: '15m'
```

Sidekiq::Scheduler config

Пример (конфигурируем)

```
# portal configuration
Sidekiq::Portal.setup! do |config|
  config.default_timezone = 'UTC' # 'UTC' by default
  config.retry_count = 3 # 0 by default
  config.retry_on = [StandardError] # [StandardError] by default

  # pre-defined sidekiq-scheduler configs (Rails example)
  config.scheduler_config = Rails.application.config_for(:sidekiq)[:schedule]

  # manual sidekiq-scheduler configs
  config.scheduler_config = {
    LoolJob: { every: '15m' },
    kek_job: { cron: '0 * * * * *', class: :KekJob }
  }
end

# global state clearing logic
RSpec.configure do |config|
  config.before { Sidekiq::Worker.clear_all }
  config.after { Timecop.return }
  config.after { Sidekiq::Portal.reload! }
end
```


Пример (тестим)

```
RSpec.describe 'HookExampleJob sheduler plan' do
  specify 'scheduled?' do
    stub_const('GLOBAL_HOOK_INTERCEPTOR', [])
    expect(GLOBAL_HOOK_INTERCEPTOR.count).to eq(0) # => true

    # do some magic 🐱
    Timecop.travel(Time.current + 2.hours)

    expect(GLOBAL_HOOK_INTERCEPTOR.count).to eq(8) # => true (🐱 magic)

    puts GLOBAL_HOOK_INTERCEPTOR # 🐱
    # => outputs:
    # 2019-12-24 03:05:39 +0300 (+15m) (Time.current from HookExampleJob#perform)
    # 2019-12-24 03:20:39 +0300 (+15m)
    # 2019-12-24 03:35:39 +0300 (+15m)
    # 2019-12-24 03:50:39 +0300 (+15m)
    # 2019-12-24 04:05:39 +0300 (+15m)
    # 2019-12-24 04:20:39 +0300 (+15m)
    # 2019-12-24 04:35:39 +0300 (+15m)
    # 2019-12-24 04:50:39 +0300 (+15m)

    end
  end
end
```

Sidekiq::Portal - Что будет дальше?

Sidekiq::Portal - Что будет дальше?

- новый режим тестирования **Sidekiq::Testing.portal!**

Sidekiq::Portal - Что будет дальше?

- новый режим тестирования **Sidekiq::Testing.portal!**
- расширение для **RSpec** (специальные rspec-матчеры)

Sidekiq::Portal - Что будет дальше?

- новый режим тестирования **Sidekiq::Testing.portal!**
- расширение для **RSpec** (специальные rspec-матчеры)
- поддержка **Sidekiq::Worker** и **Sidekiq::Cron**

Sidekiq::Portal - Что будет дальше?

- новый режим тестирования **Sidekiq::Testing.portal!**
- расширение для **RSpec** (специальные rspec-матчеры)
- поддержка **Sidekiq::Worker** и **Sidekiq::Cron**
- рандомизация порядка выполнения джобок, запланированных на одно время

Sidekiq::Portal - Что будет дальше?

- новый режим тестирования **Sidekiq::Testing.portal!**
- расширение для **RSpec** (специальные rspec-матчеры)
- поддержка **Sidekiq::Worker** и **Sidekiq::Cron**
- рандомизация порядка выполнения джобок, запланированных на одно время
- конфигурируемый **inline**-запуск отдельных джобок

Sidekiq::Portal - Что будет дальше?

- новый режим тестирования **Sidekiq::Testing.portal!**
- расширение для **RSpec** (специальные rspec-матчеры)
- поддержка **Sidekiq::Worker** и **Sidekiq::Cron**
- рандомизация порядка выполнения джобок, запланированных на одно время
- конфигурируемый **inline**-запуск отдельных джобок
- и еще много всего (смотри **ROADMAP**)



Спасибо за внимание!

Sidekiq::Portal

https://github.com/0exp/sidekiq_portal