

Stripe & Taxamo

Payments integration



Grzegorz Unijewski

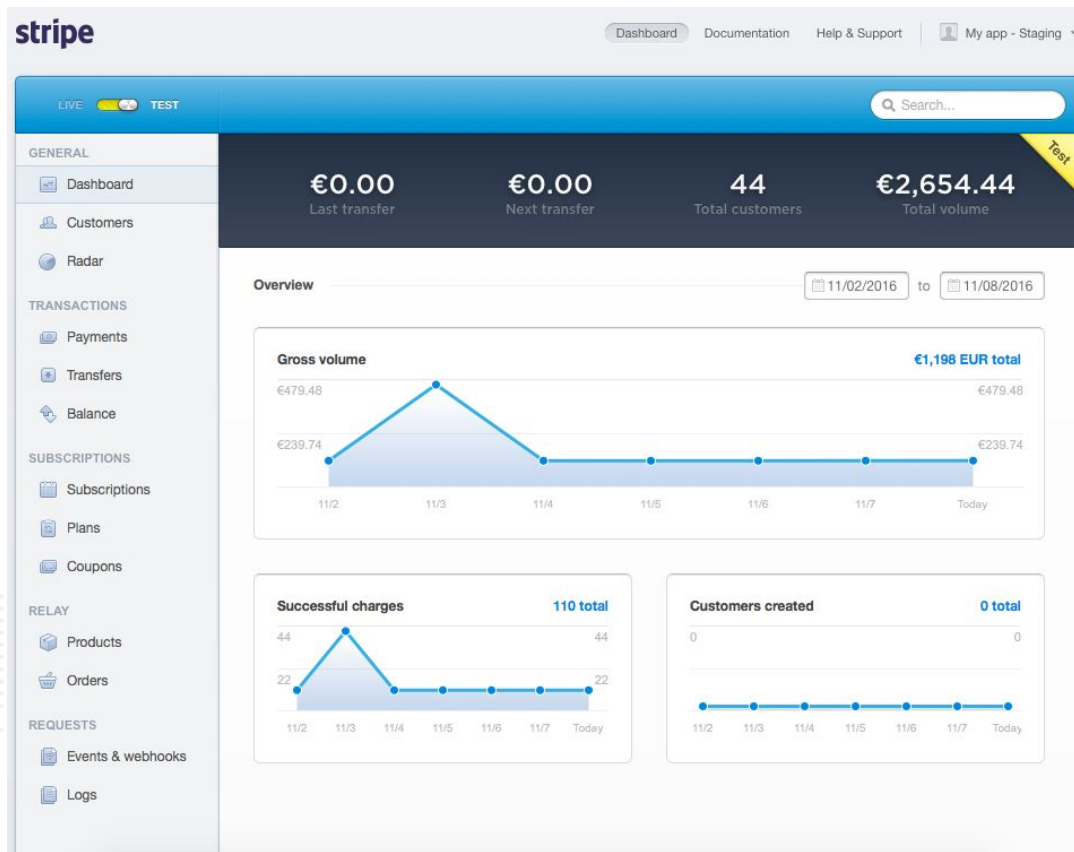
Ruby on Rails Developer

grzegorz.unijewski@netguru.co

- What is Stripe & Taxamo?
- Services integration based on my case
- Useful stuff

What is Stripe & Taxamo?

- Online payments platform
- Developer friendly documentation
- Paid per transaction
- Maintained Ruby client
- 1.4% + 20p for EU cards (2.9% for non-EU)



- Global, real-time VAT & tax solution
- Paid monthly/annually
- €25/month or €19/month (up to €10k payments/month)

taxamo

SupportIntegrate▼My account▼

TEST ☐ LIVE

Home > Dashboard

Dashboard

Transactions

Add

Upload

Tax settlement

European Union

MOSS

VIES

Domestic

Refund report


Logs

Operation logs

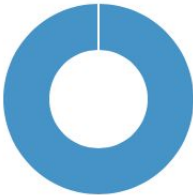
Webhook logs

Showing top 15 countries in sales in a 1-year period.

Compliance monitoring dashboard ?



Most popular regions



■ EU (100.00%)

Case: subscriptions

- Purchase
- Annulment
- Renewal

Integration - preparing

```
# Gemfile
gem 'stripe'

# config/initializer/stripe.rb
Stripe.api_key = AppConfig.stripe.api_key # secret token

# Examples:
# list charges
Stripe::Charge.list()

# retrieve single charge
Stripe::Charge.retrieve(
  "ch_123456789",
)
```

Integration - preparing

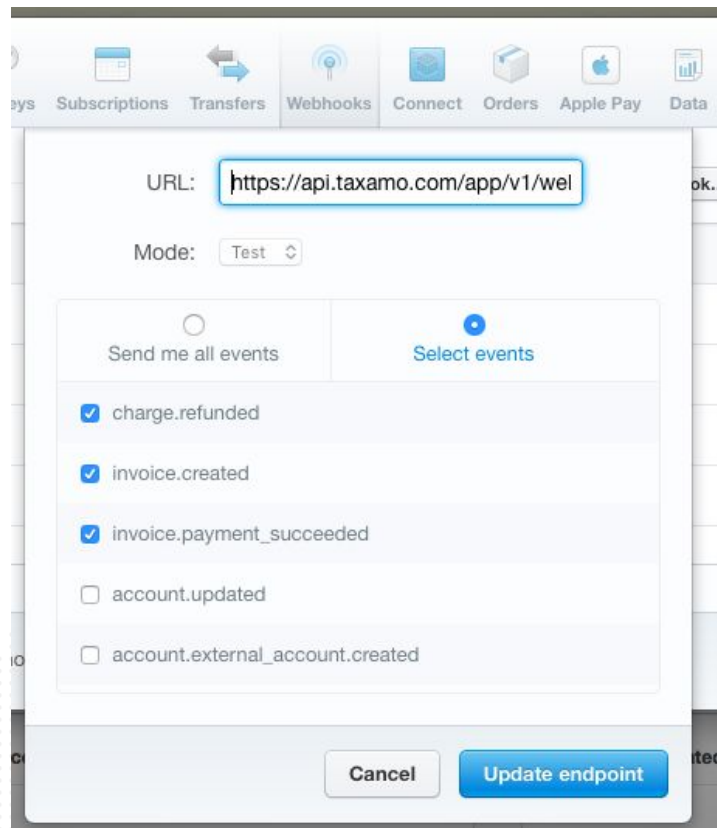
```
# Required data
:user                # resource owner
:plan                # subscription plan type
:tax_percent         # tax rate
:stripe_token        # stripe transaction key
:taxamo_token        # taxamo transaction key

# Required column in User model
create_table "users", force: :cascade do |t|
  [...]
  t.string   "stripe_cid"
end
```

Integration - preparing

URL:

`https://api.taxamo.com/app/v1/webhooks/stripe?public_token=taxamo_public_key`



The screenshot shows the 'Webhooks' configuration window in the NetGuru application. The window has a title bar with icons for various features: Payments, Subscriptions, Transfers, Webhooks (active), Connect, Orders, Apple Pay, and Data. The main content area includes a 'URL' field with the value 'https://api.taxamo.com/app/v1/webhooks/stripe?public_token=taxamo_public_key', a 'Mode' dropdown set to 'Test', and two radio buttons: 'Send me all events' (unselected) and 'Select events' (selected). Below these are five checkboxes for event types: 'charge.refunded' (checked), 'invoice.created' (checked), 'invoice.payment_succeeded' (checked), 'account.updated' (unchecked), and 'account.external_account.created' (unchecked). At the bottom right are 'Cancel' and 'Update endpoint' buttons.

URL: `https://api.taxamo.com/app/v1/webhooks/stripe?public_token=taxamo_public_key`

Mode: Test

☐ Send me all events ☒ Select events

- ☒ charge.refunded
- ☒ invoice.created
- ☒ invoice.payment_succeeded
- ☐ account.updated
- ☐ account.external_account.created

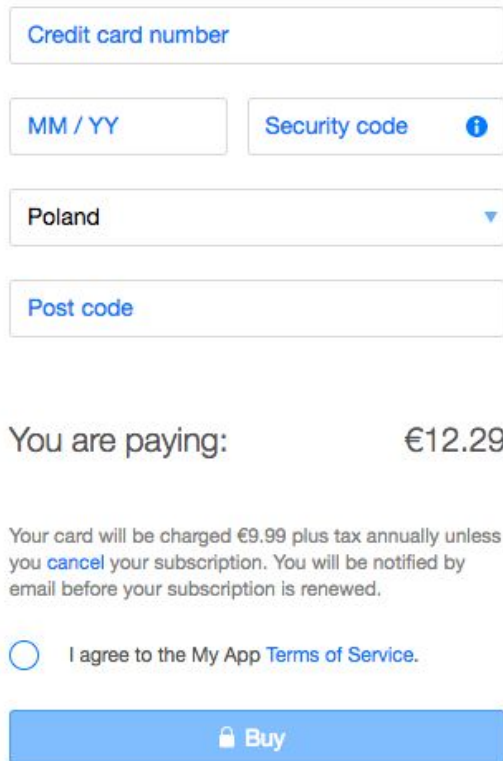
Cancel Update endpoint

Integration - preparing

For the payment form use **Stripe.js** or **ember-stripe-service**.

Examples of good confirmation data:

- Credit card's country & IP address
- Billing country & IP address
- Billing country & credit card's country



Credit card number

MM / YY Security code ⓘ


Poland ▼

Post code

You are paying: €12.29

Your card will be charged €9.99 plus tax annually unless you [cancel](#) your subscription. You will be notified by email before your subscription is renewed.

☐ I agree to the My App [Terms of Service](#).

 Buy

- Purchase
- Annulment
- Renewal

Subscription - purchase

```
# create_transaction
```

```
# delete_other_subscriptions if previous_subscriptions?
```

```
# grant_plan
```


Subscription - purchase

```
# create_transaction

## update_card if user.stripe_cid.present?
customer = Stripe::Customer.retrieve(user.stripe_cid)
customer.source = options[:stripe_token]
customer.save

## confirm_transaction
HTTParty.post(
  "https://api.taxamo.com/api/v1/transactions/#{options[:taxamo_token]}/confirm",
  { query: { private_token: AppConfig.taxamo.private_token }
})
```

Subscription - purchase

```
# create_transaction
...
## create_subscription
Stripe::Subscription.create(
  customer: user.stripe_cid,
  plan: plan,
  metadata: {
    taxamo_transaction_key: options[:taxamo_token]
  },
  tax_percent: options[:tax_percent]
)

### if we don't have customer ID
customer_id = Stripe::Customer.create(source: options[:stripe_token], email: user.email).id
user.update(stripe_cid: customer_id)
```

Subscription - purchase

```
# create_transaction
```

```
# delete_other_subscriptions if previous_subscriptions?
```

```
# grant_plan
```

Subscription - purchase

```
# delete_other_subscriptions if previous_subscriptions?  
subscriptions = Stripe::Customer.retrieve(user.stripe_cid).subscriptions  
subscriptions.to_a.drop(1).present?  
other_subscriptions_ids.map do |sub_id|  
  Stripe::Subscription.retrieve(sub_id).delete  
end  
  
# grant_plan  
account_plan = { plan: plan, currency: currency }  
SubscriptionServices::SubscriptionGrantor.new(user: user, new_plan: account_plan).call
```

- Purchase
- Annulment
- Renewal

Subscription - annulment

```
def cancel_subscription
  stripe_subscription = Stripe::Subscription.retrieve(subscription_id)
  stripe_subscription.delete(at_period_end: true)
  user.active_subscription.update(is_cancelled: true)
end
```

```
def subscription_id
  customer = Stripe::Customer.retrieve(user.stripe_cid)
  customer.subscriptions.first.id
end
```

- Purchase
- Annulment
- Renewal

Subscription - renewal

- Add a new webhook with URL:

`https://myapp.com/api/v1/subscriptions/events?access_token=your_token`

for **customer.subscription.updated** event

- Handle it in the endpoint
- Don't forget about authorization

Subscription - renewal

```
def call
  user.subscriptions.last.update(end_date: end_date)
end

private

def json_object
  @json_object ||= event_json['data']['object']
end

def user
  customer_id = json_object['customer']
  User.find_by!(stripe_cid: customer_id)
end

def end_date
  date = json_object['current_period_end']
  Time.at(date)
end
```

Useful stuff

stripe_event

```
gem 'stripe_event'

# config/routes.rb
mount StripeEvent::Engine, at: '/my-chosen-path' # provide a custom path

# config/initializers/stripe.rb
Stripe.api_key = ENV['STRIPE_SECRET_KEY'] # e.g. sk_live_1234

StripeEvent.configure do |events|
  events.subscribe 'charge.failed' do |event|
    # Define subscriber behavior based on the event object
    event.class      #=> Stripe::Event
    event.type       #=> "charge.failed"
    event.data.object #=> #<Stripe::Charge:0x3fcb34c115f8>
  end

  events.all do |event|
    events.all BillingEventLogger.new(Rails.logger)
    events.subscribe 'customer.created', CustomerCreated.new
  end
end
```

stripe-ruby-mock

```
gem 'stripe-ruby-mock', '~> 2.3.1', require: 'stripe_mock'

require 'stripe_mock'

describe MyApp do
  let(:stripe_helper) { StripeMock.create_test_helper }
  before { StripeMock.start }
  after { StripeMock.stop }

  it "creates a stripe customer" do
    # This doesn't touch stripe's servers nor the internet!
    # Specify :source in place of :card (with same value) to return customer with source data
    customer = Stripe::Customer.create({
      email: 'johnny@appleseed.com',
      card: stripe_helper.generate_card_token
    })
    expect(customer.email).to eq('johnny@appleseed.com')
  end
end
```

stripe-ruby-mock

```
stripe_helper.create_plan(my_plan_params)
stripe_helper.delete_plan(my_plan_params)
stripe_helper.generate_card_token(my_card_params)
```

```
let(:customer) do
  Stripe::Util.convert_to_stripe_object(
    {
      object: 'customer',
      source: 'source',
      subscriptions: subscriptions_array
    }, {}
  )
end

let(:stripe_subscription) do
  Stripe::Util.convert_to_stripe_object({ object: 'subscription', id: '1' }, {})
end
```

- Stripe API reference: <https://stripe.com/docs/api/ruby>
- Taxamo API reference: <https://www.taxamo.com/apidocs/>
- Stripe-ruby: <https://github.com/stripe/stripe-ruby>
- Stripe-ruby-mock: <https://github.com/rebelidealist/stripe-ruby-mock>
- Stripe_event: https://github.com/integrallis/stripe_event
- Taxamo-ruby: <https://github.com/taxamo/taxamo-ruby>
- Stripe JS reference: <https://stripe.com/docs/stripe.js>
- Ember-stripe-service: <https://github.com/code-corps/ember-stripe-service>



Q&A



Ruby on Rails Development Team

- ~ 80 developers including
 - ~ 20 seniors
 - ~ 10 team leaders

Ruby on Rails Developer must-haves:

- Have a hands-on knowledge of Ruby on Rails, HAML/SASS
- Have knowledge of at least one of JavaScript frameworks (AngularJS/Ember.js, React, jQuery)
- You know how to deal efficiently with databases and various queries
- You have very good command of written and spoken English (minimum B2)
- You have experience in direct communication with clients

Juniors are more than welcome! You just need to:

- Have some basic knowledge in Ruby and Ruby on Rails
- Have at least one Rails app on your GitHub account
- Be fluent in English - at least B2 level, spoken and written
- Be ready to work at least 35 hours/week, from Monday till Friday

How to join our awesome team?

- **Step one** - check our career path and see what level you are
netguru.co/career/paths
- **Step two** - apply for the proper position - netguru.co/career
- **Step three** - take part in our recruitment process - solve the task and join us for the remote interview and pair programming
- **Step four** - welcome onboard!

Don't hesitate and join our team!

Any questions regarding recruitment process?

Drop us a line at **jobs@netguru.co**

Thanks!