Version 1
check for updates!

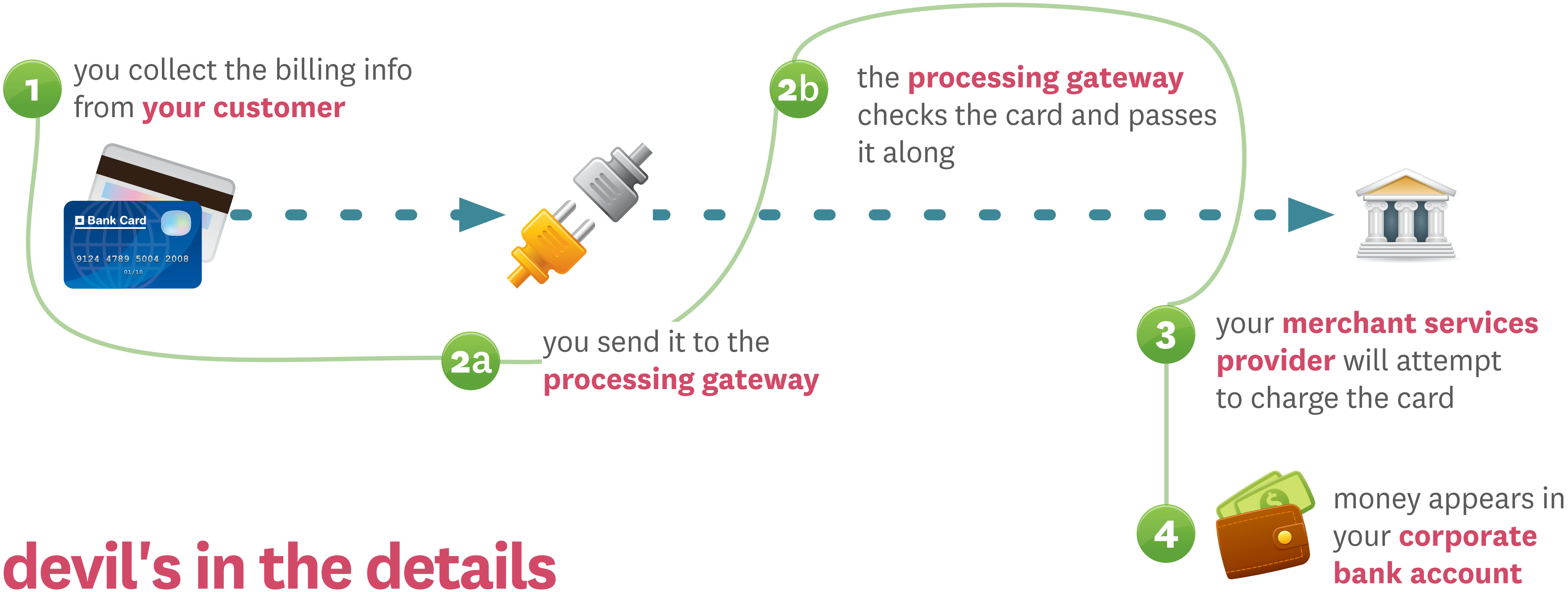
jump start credit card processing

sponsored by freckle time tracking

by amy hoy, thomas fuchs &
dieter komendera

card processing lifecycle: 10,000 ft view

freckle
time tracking rethought



1 you collect the billing info from **your customer**

2a you send it to the **processing gateway**

2b the **processing gateway** checks the card and passes it along

3 your **merchant services provider** will attempt to charge the card

4 money appears in your **corporate bank account**

devil's in the details

1 You decide **how much data** to collect; in reality, only the **card number** and **expiration date** are truly required

2a You'll use an **API** over a **secure HTTPS connection** to talk to your **gateway**; code your own interface or use any number of handy libraries

2b **Address Verification Service (AVS)** happens here, if you use it.

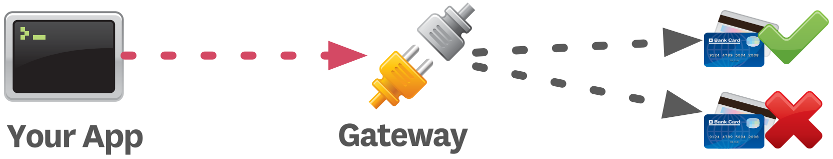
card processing lifecycle: key actions

freckle
time tracking rethought

authorize

```
auth = @gateway.authorize(money_in_cents, card_obj, options) #activemerchant
```

You attempt to place a hold on the credit card. If successful, you can either continue the charge or the hold will expire after a period of time.



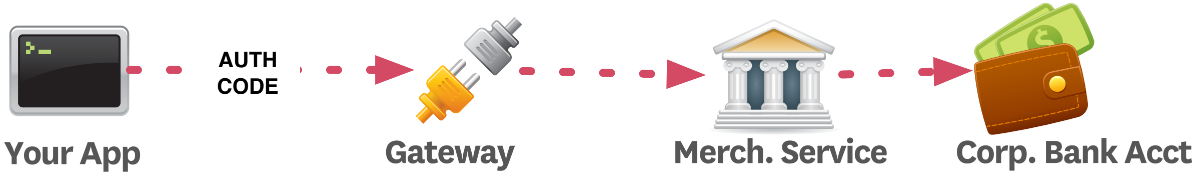
Hold succeeded! You receive an **authorization code**.

Hold failed!

capture

```
@gateway.capture(money_in_cents, auth, options) #activemerchant
```

You finalize the hold—you "capture" the money. You supply the **authorization code** to complete the transaction.



purchase

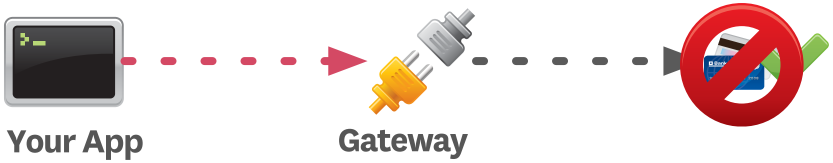
```
@gateway.purchase(money_in_cents, cc_object, options) #activemerchant
```

authorize + **capture** in one request

void

```
@gateway.void(auth, options) #activemerchant
```

Kill a successful hold, instead of waiting days for it to expire.



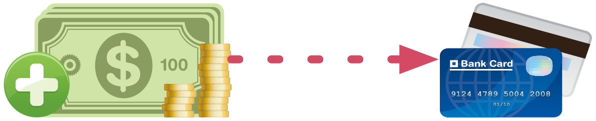
Hold removed!

other actions

transactions

credit

You return money to / place money on the provided credit card. A credit, rather than a debit.



managing data

Storing credit cards securely is a major hassle. In the US, you'll have to comply with very stringent security requirements before the credit card banks will allow you to do it. It's much easier to let your **processing gateway** do it for you—they're the experts.

store

Store credit card details (number, expiration date, billing address) for a new customer.

update

Update credit card details (number, expiration date, billing address) for an existing customer.

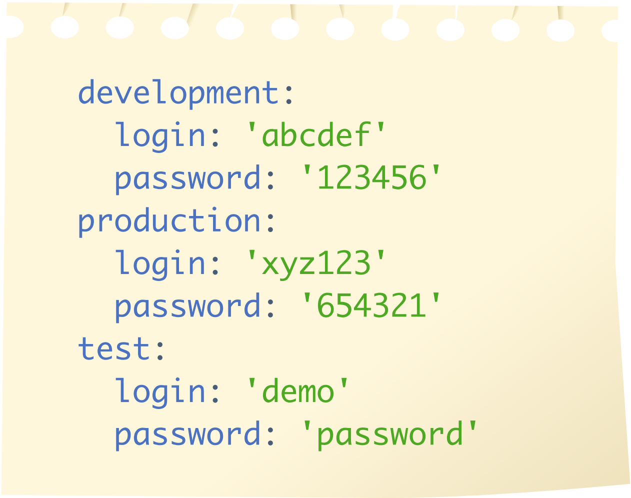
using activemerchant with ruby

activemerchant rocks

activemerchant is by far the most popular way of handling any kind of credit card transactions with Ruby and Ruby on Rails.

To get started with activemerchant:

1. Check the supported gateways list (linked right) to be sure you've got / will be using one of the many supported credit card processing gateway services.
2. Download and install **activemerchant** as a rubygem (recommended) or a Rails plugin (instructions linked right).
3. Configure your **gateway.yml** file, like so:



```
development:
  login: 'abcdef'
  password: '123456'
production:
  login: 'xyz123'
  password: '654321'
test:
  login: 'demo'
  password: 'password'
```

online resources

- **supported gateways**
- **how to install activemerchant**
- **github repository**
- **peepcode book** (recommended!)
- **SaaS railskit** (recommended!)

using activemerchant with ruby

freckle
time tracking rethought

4 Enter the Ruby interactive console (**irb**). Type:

```
require 'rubygems'
require 'active_merchant'
```

5 Set **activemerchant** to test mode:

```
ActiveMerchant::Billing::Base.mode = :test
```

creating activemerchant objects

```
@creditcard =
ActiveMerchant::Billing::CreditCard.new({
       :number => '4111111111111111',
       :year => 2010,
       :month => 1,
       :verification_value => '123',
       :type =>'visa',
       :first_name => 'John',
       :last_name => 'Doe' })
```

```
@gateway =
ActiveMerchant::Billing::Base.gateway('authorize_net')
.new(config_from_file('gateway.yml'))
```

using activemerchant with ruby

freckle
time tracking rethought

6 Create a new gateway, new credit card, and create a test charge and then void it (remember, you should be in dev mode!)

```
require 'rubygems'
require 'active_merchant'

ActiveMerchant::Billing::Base.mode = :test

@gateway =
ActiveMerchant::Billing::Base.gateway('authorize_net')
.new(config_from_file('gateway.yml'))

@creditcard =
ActiveMerchant::Billing::CreditCard.new({
       :number => '4111111111111111',
       :year => 2010,
       :month => 1,
       :verification_value => '123',
       :type =>'visa',
       :first_name => 'John',
       :last_name => 'Doe' })

response = @gateway.authorize(100, @credit_card)
response.success? || @gateway.void(response.authorization).success?
```

Remember, all "money" is in cents!
$1.00 = 100

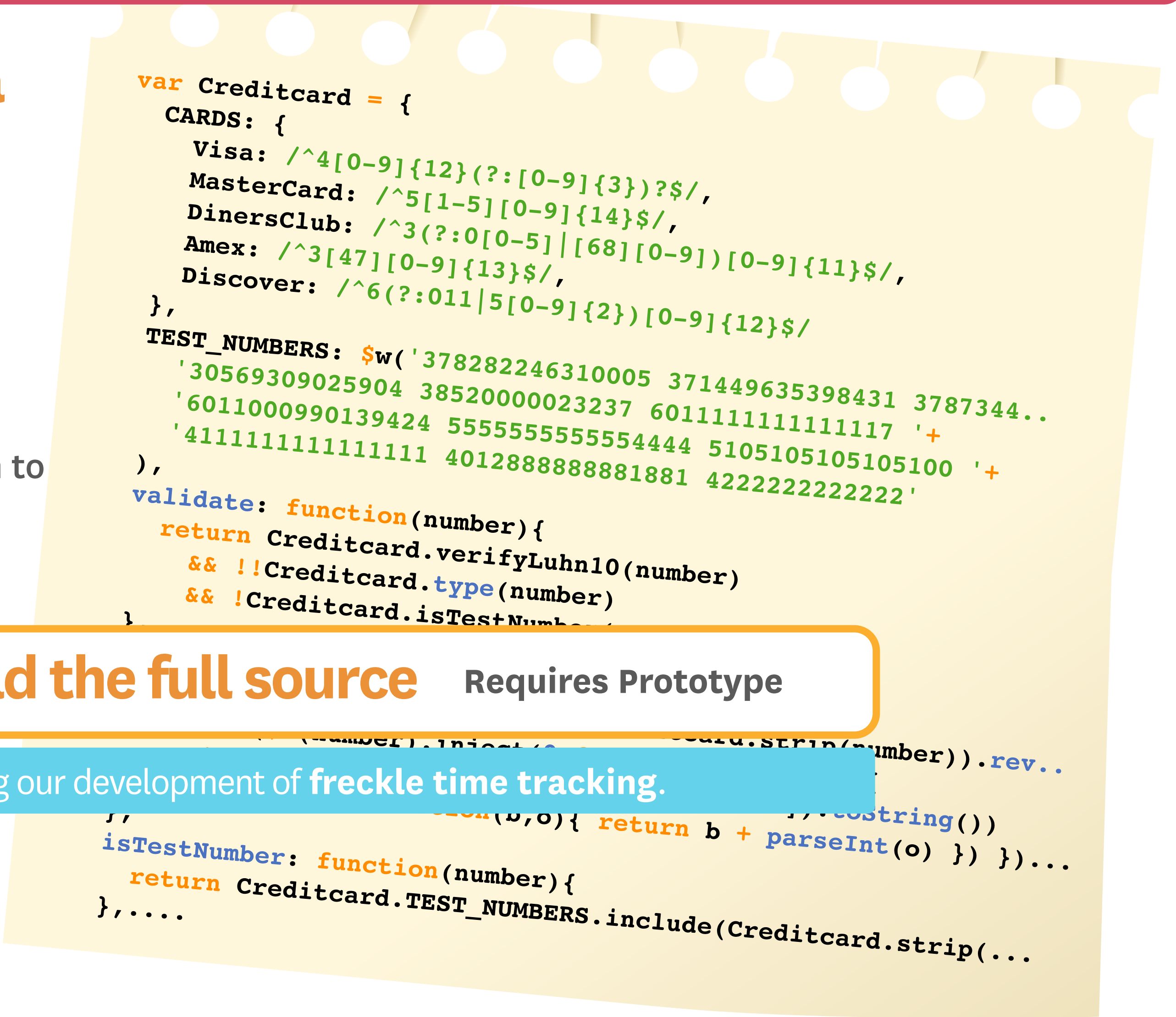
JavaScript card detection & validation

freckle
time tracking rethought

Pre-process Card Data

This script by Thomas Fuchs:

- ✓ detects card types (Visa, etc.)
- ✓ detects test card numbers
- ✓ validates card numbers using **Luhn10 checksums**
- ✓ has a handy `strip()` function to remove white space & dashes



```
var Creditcard = {
  CARDS: {
    Visa: /^4[0-9]{12}(?:[0-9]{3})?$/,
    MasterCard: /^5[1-5][0-9]{14}$/,
    DinersClub: /^3(?:0[0-5]|[68][0-9])[0-9]{11}$/,
    Amex: /^3[47][0-9]{13}$/,
    Discover: /^6(?:011|5[0-9]{2})[0-9]{12}$/
  },
  TEST_NUMBERS: $w('378282246310005 371449635398431 3787344..
    '30569309025904 38520000023237 6011111111111117 '+
    '6011000990139424 5555555555554444 5105105105105100 '+
    '4111111111111111 4012888888881881 4222222222222'
  ),
  validate: function(number){
    return Creditcard.verifyLuhn10(number)
      && !!Creditcard.type(number)
      && !Creditcard.isTestNumb
  },
      (number).inject(          card.strip(number)).rev..
                                 }).toString())
                ion(b,o){ return b + parseInt(o) }) })...
  },
  isTestNumber: function(number){
    return Creditcard.TEST_NUMBERS.include(Creditcard.strip(...
  },....
```

Download the full source Requires Prototype

This library was created during our development of **freckle time tracking.**

activemerchant peepcode PDF

PeepCode

Active Merchant

Show me the money!

by Cody Fauser

click!

70 amazing pages, absolutely packed with information on activemerchant. We read it cover to cover and it helped us tremendously, in the way that API docs never can.

And it's only $9!

Software as a Service (SaaS) Rails Kit

RailsKits

click!

The **SaaS Rails Kit** is a combination library, application code & data model setup that helps you get a **SaaS app off the ground in no time**.

It may sound expensive at $249, but we estimate that it **saved us at least 20-25 hours**. At our billing rate, that's about **$2,100 to $2,600**. And it helped us launch **freckle** at least a week sooner. To say we're delighted with the savings... well, it's an understatement.

I bought both the peepcode PDF & SaaS Rails Kit with my own money, and recommend them unreservedly.

In the interest of full disclosure: I became a Rails Kit affiliate because I was so pleased with the SaaS RK.

sponsored by freckle time tracking