



**Version 1**  
check for updates!

this quick crash  
course will get you  
started



# jump start credit card processing

sponsored by  **freckle time tracking**

by amy hoy, thomas fuchs &  
dieter komendera



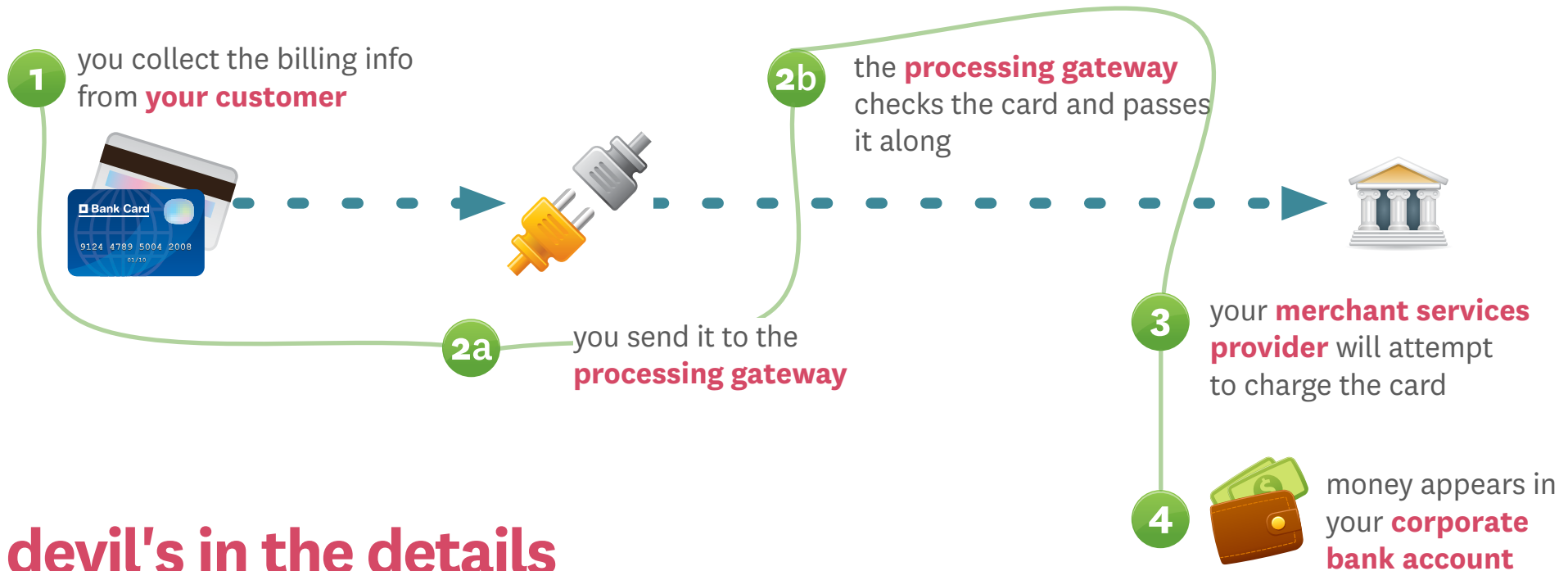
**Version 1**  
check for updates!

# **PART 1:** How it Works Flow & Terms

sponsored by  **freckle time tracking**

*by amy hoy, thomas fuchs &  
dieter komendera*

# card processing lifecycle: 10,000 ft view



## 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

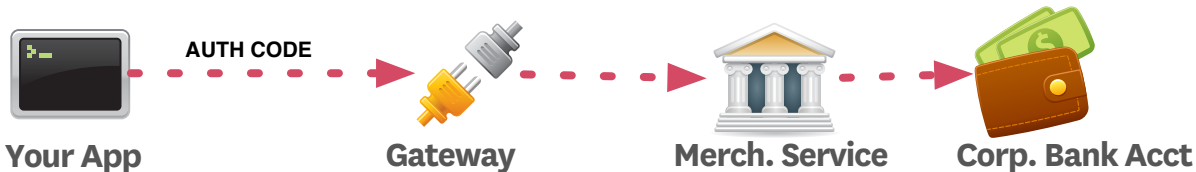
**authorize** `auth = @gateway.authorize(money_in_cents, card_obj, options) #activemerchant`

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.



**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.



## 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.



# three ways to validate cards

## Checksum verification

Checksum verification checks the likelihood that the credit card number is **real** by means of an algorithm called Luhn10. But, this doesn't mean it's a **usable** card. However, it's a good first defense.

WHEN:   you **collect** the card

RELIABILITY:  **LOW**

INVASIVENESS:  **NON-INVASIVE**

YOUR COST:  **NONE**

## Address Verification (AVS)

AVS is meant to **check the billing address** provided against the address the credit card company has. However, it's not useful for **non-US customers**, and it's very typical for a **genuine card owner** to enter information that is slightly incorrect. AVS is, therefore, not the silver bullet it's meant to be.

WHEN:   you **submit** the card to the **processing gateway**


RELIABILITY:  **LOW**

INVASIVENESS:  **NON-INVASIVE**

YOUR COST:  **FEEES MAY APPLY**

## Test charge *and/or* real charge

If you want to verify a card **for later billing**, your best bet is to perform a test charge: charge a small amount (ideally \$1.00) to the card, & if it comes back OK, **void the transaction**. If you'll be charging the customer immediately, & you're in a **low-fraud market**, the best way to ensure a card can be charged is to charge it.

WHEN:   you **charge** a token amount, & reverse it

RELIABILITY:  **HIGH**

INVASIVENESS:  **INVASIVE**

YOUR COST:  **FEEES MAY APPLY**



**Version 1**  
check for updates!

# **PART 2:** ActiveMerchant & JavaScript

sponsored by  **freckle time tracking**

*by amy hoy, thomas fuchs &  
dieter komendera*



## 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

- 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

## 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 601111111111117 '+'  
    '6011000990139424 5555555555554444 5105105105100 '+'  
    '411111111111111 4012888888881881 422222222222'  
  ),  
  validate: function(number){  
    return Creditcard.verifyLuhn10(number)  
      && !!Creditcard.type(number)  
      && !Creditcard.isTestNumber(  
        Creditcard.strip(number)).rev..  
      ),  
  },  
  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**.



**Version 1**  
check for updates!

# **PART 3:** Getting Your Accounts

sponsored by  **freckle time tracking**

*by amy hoy, thomas fuchs &  
dieter komendera*



## corporate bank account

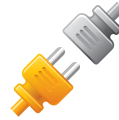
Your corporate bank account is **where your money will go after credit card transactions have cleared**. You'll specifically want a corporate account; it will be difficult to sign up for a merchant account without one.



## merchant bank account

The merchant bank account is a confusing beast. It is a bank account, but not one you can ever access directly.

The merchant bank account is **where the actual credit card transactions occur**, and the merchant services provider is the company that holds the agreements with the credit card companies themselves (Visa, Mastercard, Amex, etc.)



## cc processing gateway

Gateways serve three purposes:

- ✓ they offer address verification (AVS)
- ✓ they pass on your CC processing requests to the merchant bank service, acting as your interface, and
- ✓ they will store your customer's credit cards in a secure manner which you really don't want to try and implement on your own.

In some cases, you can get extra services from your processing gateway, such as eCheck processing (using the account numbers on the bottom of checks) and automated recurring billing (e.g. monthly billing).

## in preparation

- ☐ **incorporate.**  
Forming an LLC is a good idea for any business endeavor. We used & would recommend HBS ([delawareinc.com](https://delawareinc.com)) to incorporate in Delaware. If you cannot form an LLC, file for a sole proprietorship license from your state and/or county.
- ☐ **file for an Employer Identification Number (EIN).**  
The EIN, or TIN (Tax ID Number), etc., is essentially a Social Security Number for your new business. Some incorporation services (like HBS) will do this for you.
- ☐ **apply for a corporate bank acct.**  
It's easiest to do this in person at a local branch: take your letters of incorporation, your EIN/TIN proof, proof of address, and govt-issued photo ID with you (passport is best).

## merchants & gateways

- ☐ **apply for a merchant account.**  
Merchant accounts vary little in terms of features, so you'll be comparing mainly on basis of price and service. Be sure to ask for a table of all fees, requirements for acceptance, and which card types are included. Your local bank may be a simple choice.  
  
You will need your letters of incorporation, EIN/TIN proof, govt-issued photo ID, proof of address, and a bank letter or canceled check.
- ☐ **apply for a credit card processor.**  
The big 2 available for small business are TrustCommerce and Authorize.net. Compare based on the friendliness of their APIs & documentation, special features like recurring billing, schedule of fees, rates, and customer service.  
  
You'll need all of the same paperwork you've been accumulating, plus your merchant account information.

# activemerchant peepcode PDF



**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!**

excellent  
resources we use  
and recommend



## Software as a Service (SaaS) Rails Kit



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.