

# How Braintree Builds a Platform for Developers

Paul Gross

[paul.gross@braintreepayments.com](mailto:paul.gross@braintreepayments.com)

[twitter.com/pgr0ss](https://twitter.com/pgr0ss)

[github.com/pgr0ss](https://github.com/pgr0ss)

[pgrs.net](http://pgrs.net)

# Braintree

Braintree is a payment gateway

A payment gateway is software that allows merchants to process credit card payments from your website and/or application

# Our Merchants



# Our Merchants

We love technically savvy companies

We target developers

# Developers

We love developers

We target our product at developers

"We want to be the undisputed, #1 preferred payments provider for developers"

# Why?

Developers are often the people choosing payment gateways

Developers are vocal - Meetups, Conferences, Twitter, Google+

Developers help make our products better - Open source, 3rd party products

# How We Target Developers

Make it easy for developers

Build developer specific features

Build and foster 3rd party tools

Engage the developer community

**Make it Easy for Developers**



# Client Libraries

## API Docs - Languages



## API Docs - Mobile Platforms



## Client-side Encryption



# Handle the Annoying Stuff For Developers

SSL verification

Request timeouts

Building request XML

Parsing responses

Error code constants

## Follow Local Idioms

```
# ruby
result = Braintree::Transaction.sale(
  :amount => "100.00",
  :credit_card => {
    :number => "5105105105105100",
    :expiration_date => "05/12"
  }
)

// java
TransactionRequest request =
  new TransactionRequest()
    .amount(new BigDecimal("1000.00"))
    .creditCard()
      .number("4111111111111111")
      .expirationDate("05/2009")
      .done();

Result<Transaction> result =
  gateway.transaction().sale(request);
```

# Backwards Compatibility

Once a developer integrates, we don't want them to have to change

We constantly release, but we always ensure backwards  
compatibility

# Builds

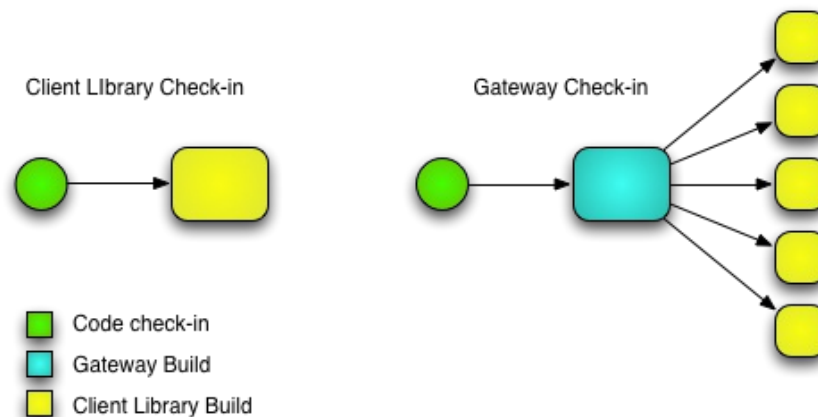
Each client library has a build

This build runs the tests on the master branch of the client library  
against the master branch of our gateway code

# Triggers

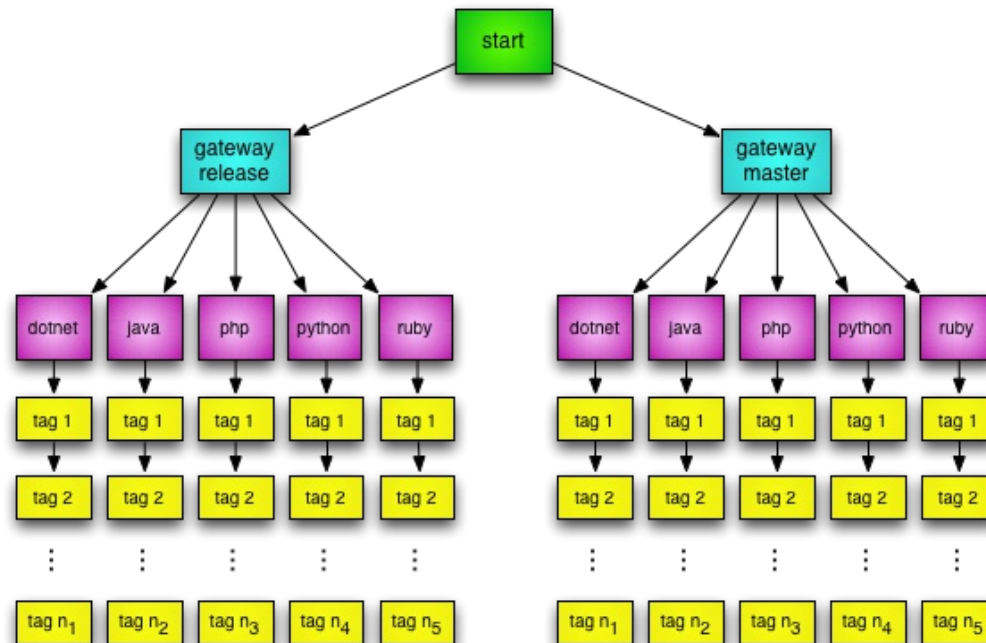
Code is pushed to the master branch of a client library

Code is pushed to the master branch of the gateway



# Backwards Compatibility

We test every released version of every client library each night



# Developer Specific Features

Client-side encryption

Webhooks

Advanced Search API



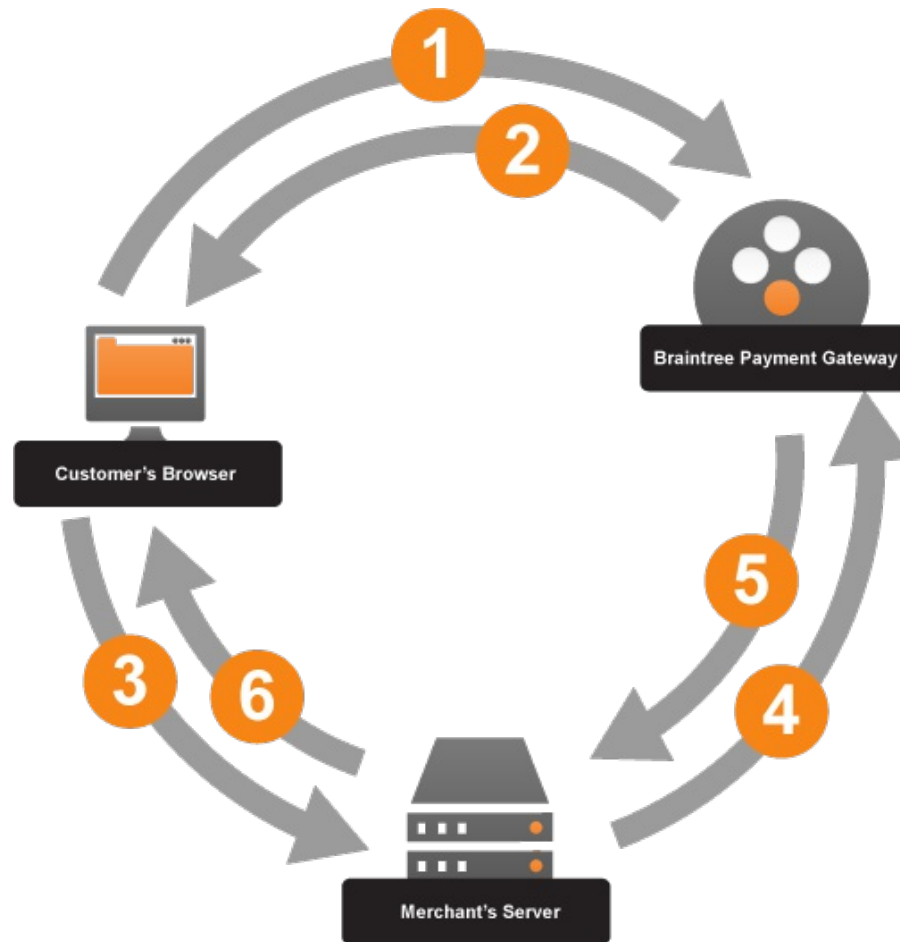
# The Problem

Merchants don't want credit card data passing through their servers

Traditional solution is Transparent Redirect

Credit card data is sent directly to Braintree, not to the merchant

# Transparent Redirect



# Transparent Redirect Problems

The entire form is POSTed to Braintree

Difficult to do custom validations

Hard to do one page signup forms

# Client-side Encryption

Encrypt the data in the browser with javascript and asymmetric encryption

POST the form to the merchant

Pass along required fields to Braintree

## Example

```
// javascript
var braintree = Braintree.create("encryption-key");
var encNum = braintree.encrypt("4111111111111111");
var encExpiration = braintree.encrypt("01/2014");
```

```
# ruby
result = Braintree::Transaction.sale(
  :amount => "100.00",
  :credit_card => {
    :number => encrypted_credit_card_number,
    :expiration_date => encrypted_expiration_date
  }
)
```

# Webhooks

Software as a service companies need to suspend service when  
billing fails

One method is to poll our system to find state of subscription

Requires a background job or real time querying

# Webhooks

Webhooks notify merchants when something happens to the subscription

Developers give us a webservice endpoint, and we call it

Simpler and less work for developers

# Advanced Search API

Traditionally, someone would have to log into our control panel for advanced searching

We give developers an easy way to integrate advanced searching into their applications



```
# ruby
results = Braintree::Transaction.search do |search|
  search.order_id.is "myorder"
  search.customer_first_name.starts_with "Tim"
  search.shipping_street_address.contains "Main St"
end
```

```
// java
TransactionSearchRequest request =
  new TransactionSearchRequest().
    orderId().is("myorder").
    customerFirstName().startsWith("Tim").
    shippingStreetAddress().contains("Main St");
```

```
ResourceCollection<Transaction> collection =
  gateway.transaction().search(request);
```

# 3rd Party Tools

Not everyone writes their system from scratch

Many developers start with a shopping cart system

We've already integrated with many of these so developers don't  
have to

# 3rd Party Tools



# We Engage the Developer Community

Open source

Conferences

Meetups

# Open Source

We use a lot of open source software

We contribute back

We release software that we wrote

# Curator

Model and repository framework for Ruby (not ActiveRecord)

[github.com/braintree/curator](https://github.com/braintree/curator)

# Supply Drop

Automation for server setup with puppet and capistrano

[github.com/pitluga/supply\\_drop](https://github.com/pitluga/supply_drop)

# Jukebox2

Communal music player for team environments

[github.com/thehammer/jukebox2](https://github.com/thehammer/jukebox2)

The screenshot displays the Jukebox2 web interface. At the top, a dark navigation bar contains the site name 'jukebox2', menu items 'Add', 'Stats', 'Users', and 'Hammertimes', a green 'SIGN UP' button, a search bar with the placeholder 'Search for Artist, Album or Song', and a user profile link 'who are you?'. The main content area features a large album cover for '100th Window' by Massive Attack. To the right of the cover, the song title 'What Your Soul Sings' is displayed in a large font, followed by the artist 'Massive Attack' and the album name '100th Window'. Below this, the duration '6:40' and a 'Play' button are visible. On the far right, statistics are shown: 'Play count: 1', 'Skip count: 0', 'Owner: test', and 'Requester: (randomizer)'. Underneath the main player, a section titled 'Playing Music From' shows two user avatars. Below that, a 'Playlist' section lists two items: '1. Words I Never Said (Feat. Skylar Grey) [Explicit] Lupe Fiasco' and '2. Sall Awolnation', each with 'Requester: test' and 'Owner: test' information.



# vim dotfiles

All of our vim settings

[github.com/braintreeps/vim\\_dotfiles](https://github.com/braintreeps/vim_dotfiles)

# Conclusion

We love developers

We try to make their lives easier

We engage in the community

They advocate for us

Questions?

