Practices of a Modern Tech Startup

Paul Hinze
phinze

Braintree™

The Most Scariest Things

(for developers)

1st

2nd


Braintree™

don't you worry your little head

Braintree™

we'll take the scary stuff

What We Do

rain tree

You
ROVIO
Bank
Bank
Bank
Bank

Clean APIs

Bank Integration

Secure Storage

Clean APIs

Today

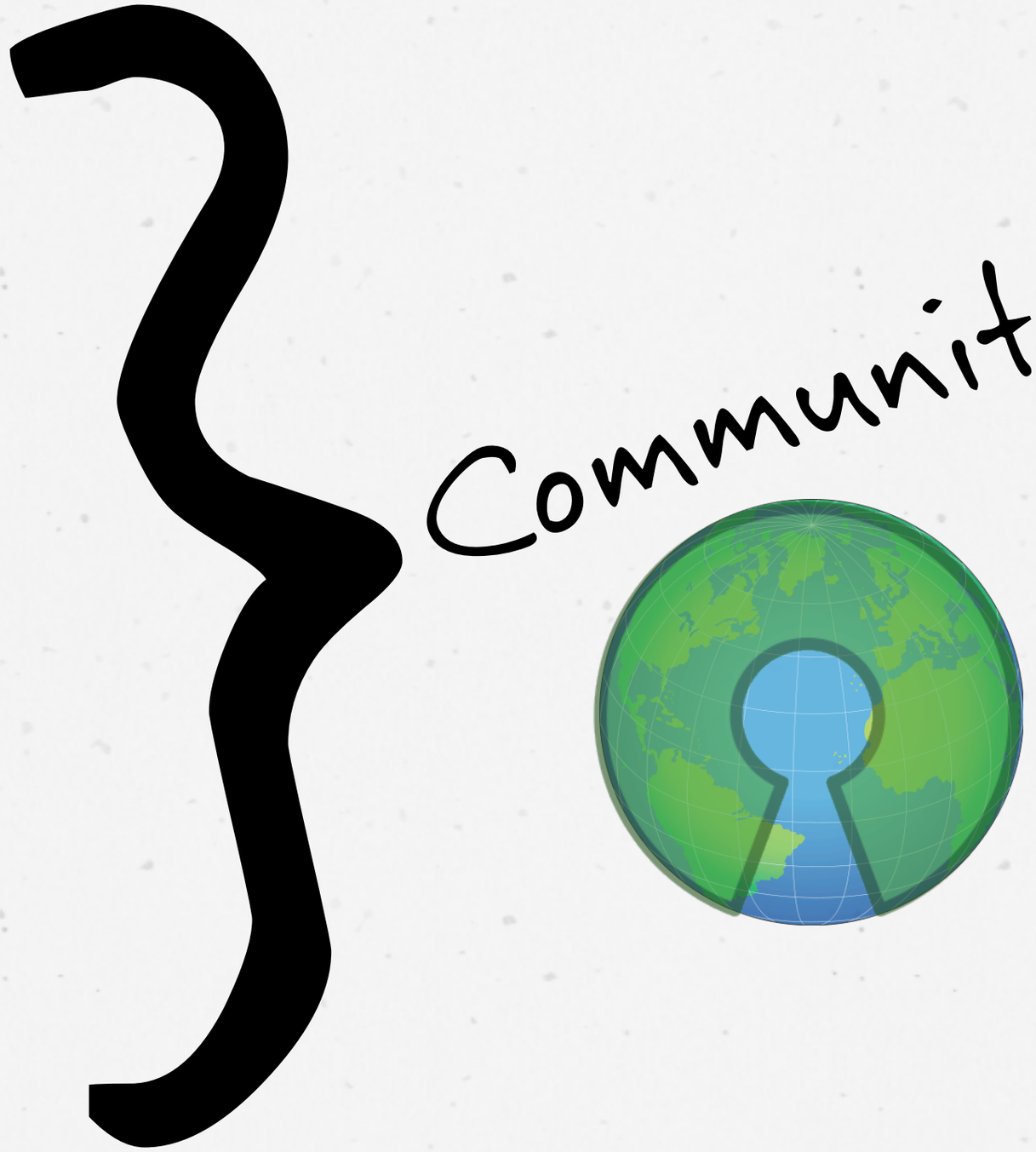
Team

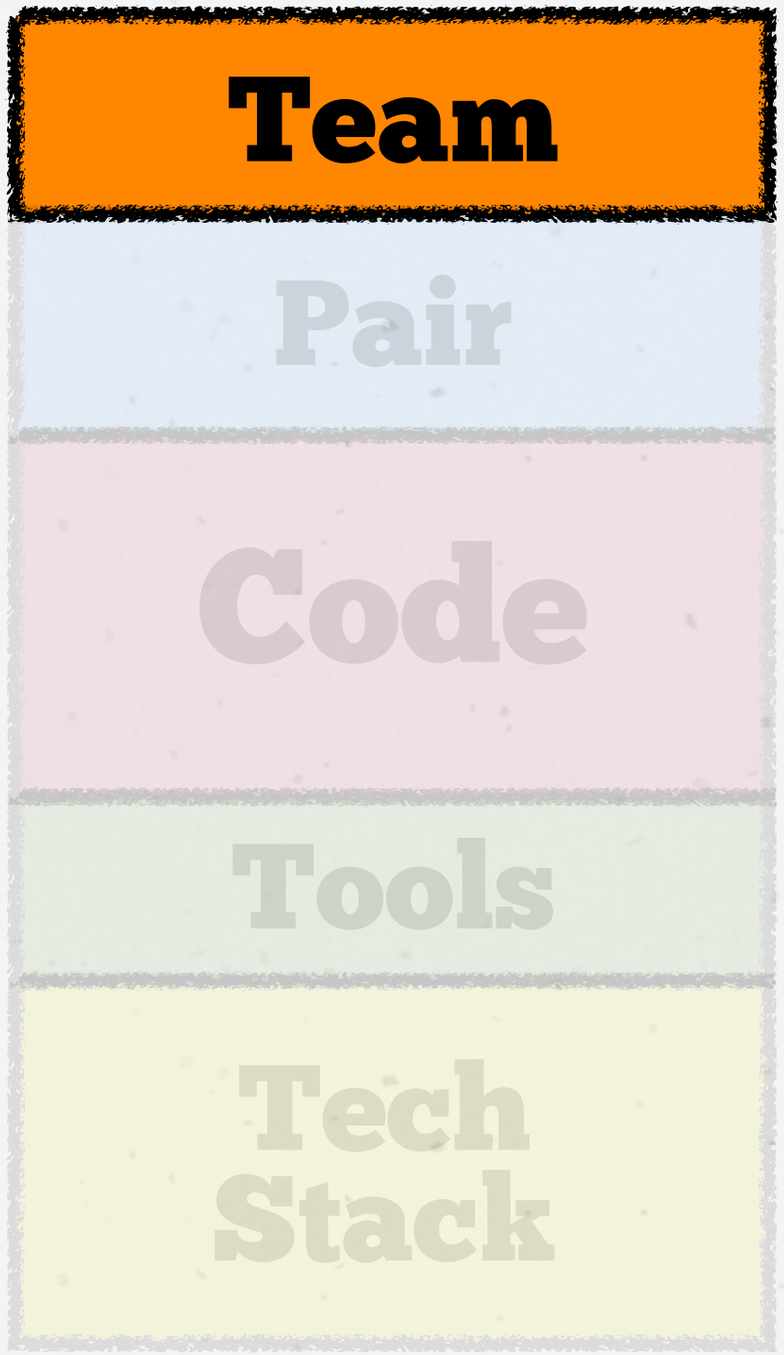
Pair

Code

Tools

Tech Stack





Agile

Requirements,

then **Design**,

then **Implementation**,

then **Verification**,

then **Maintenance**.

months to years

Requirements,

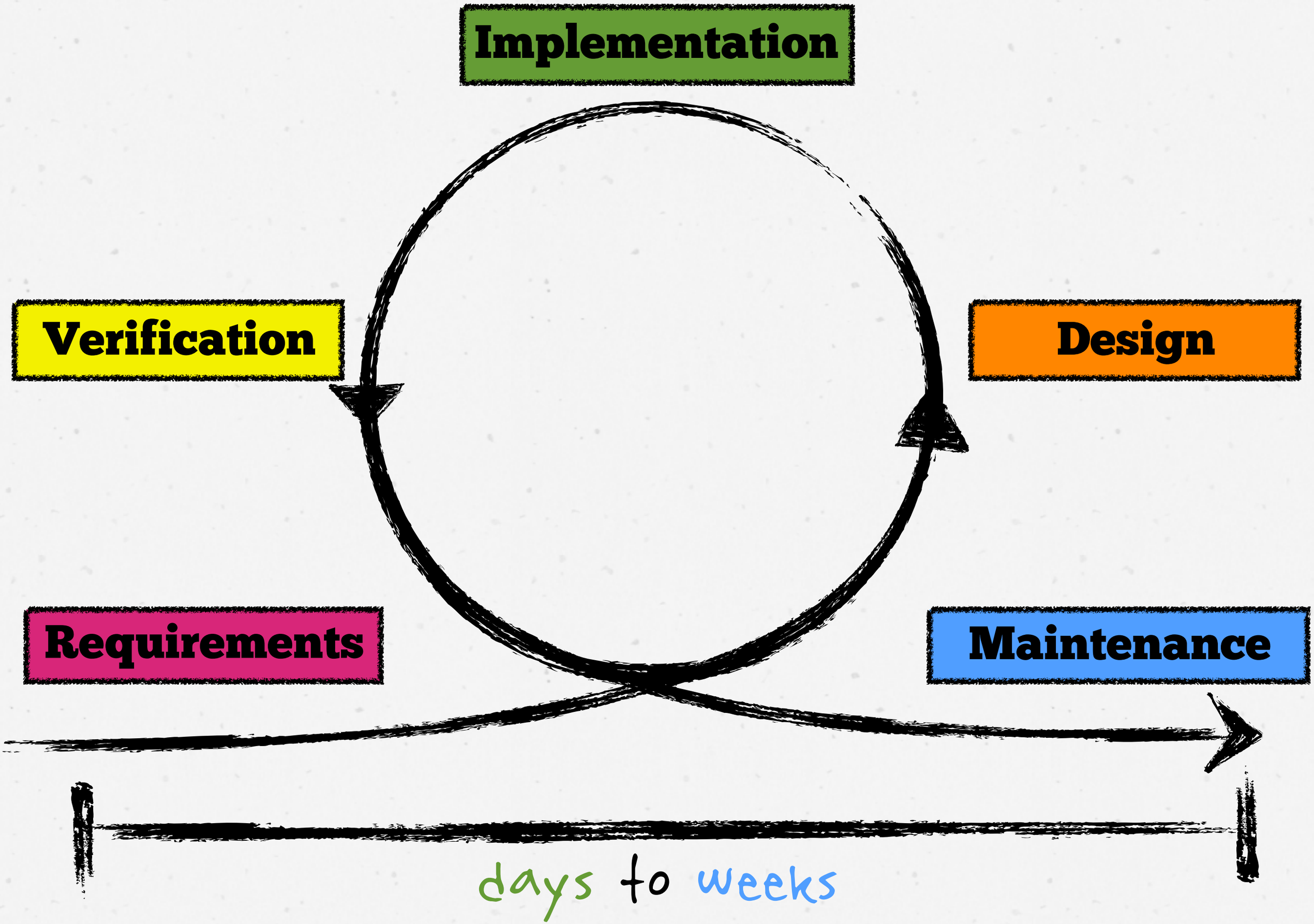
then **Design**,

then **Implementation**

then [illegible],

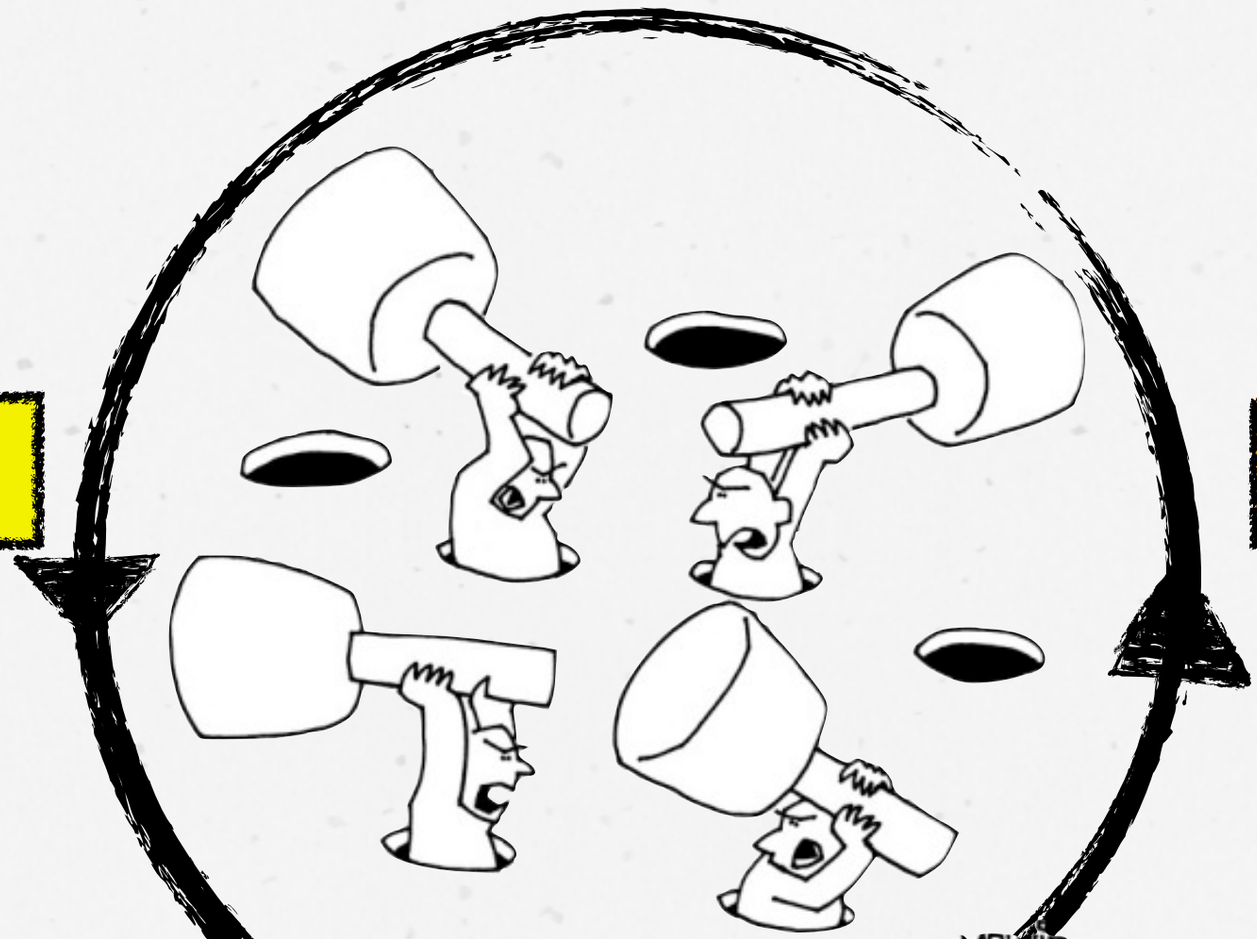
then **Mai[illegible]**.

months to years


Implementation
Verification
Design
Requirements
Maintenance
days to weeks

Implementation

Verification

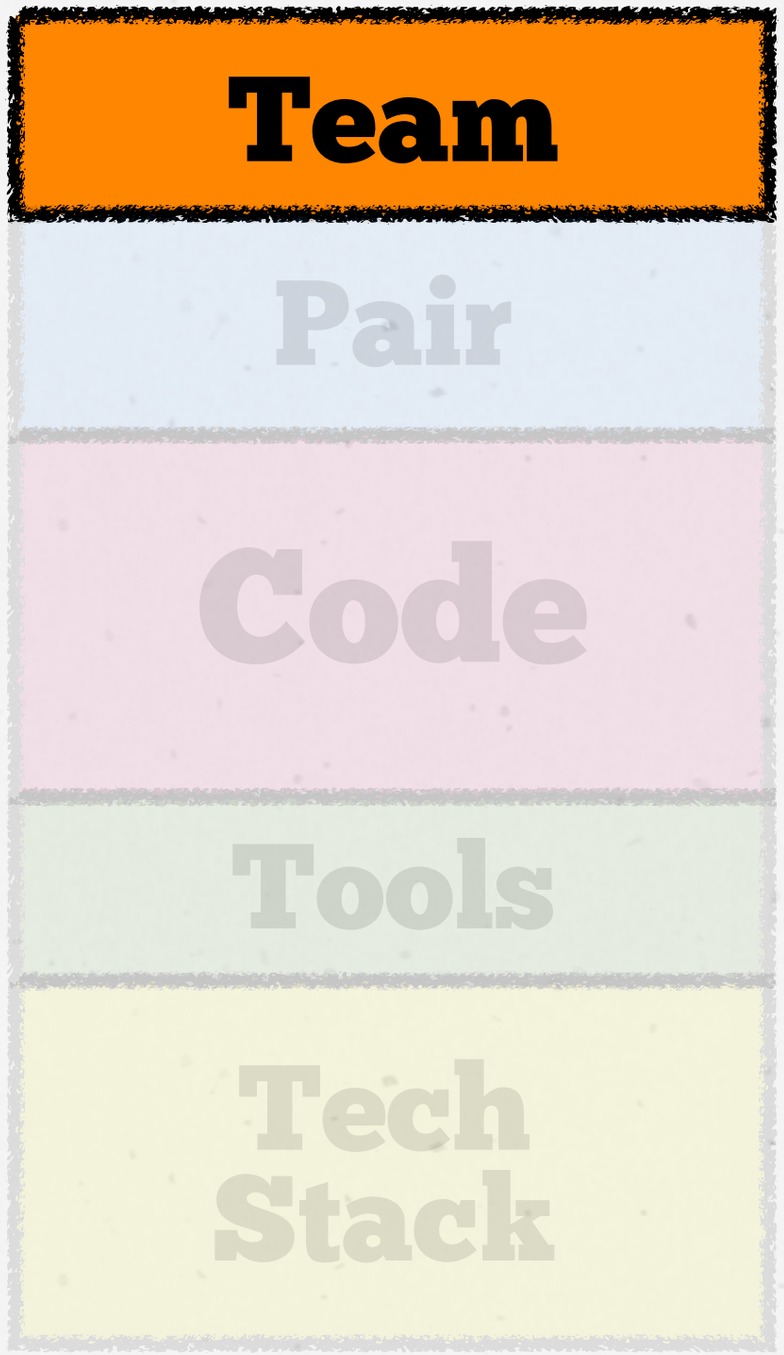


Design

Requirements

Maintenance

days to weeks



Agile

in practice

Team

Pair

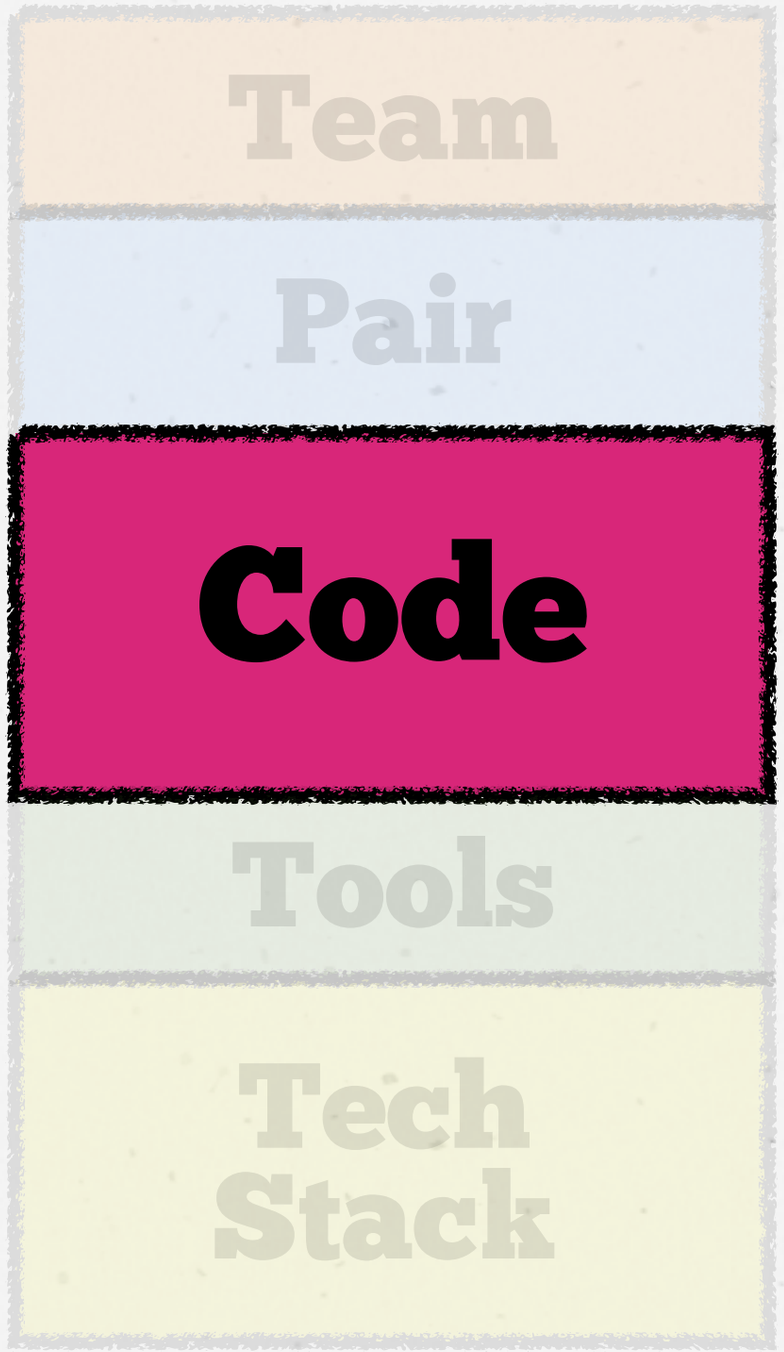
Code

Tools

Tech Stack

Pair Programming

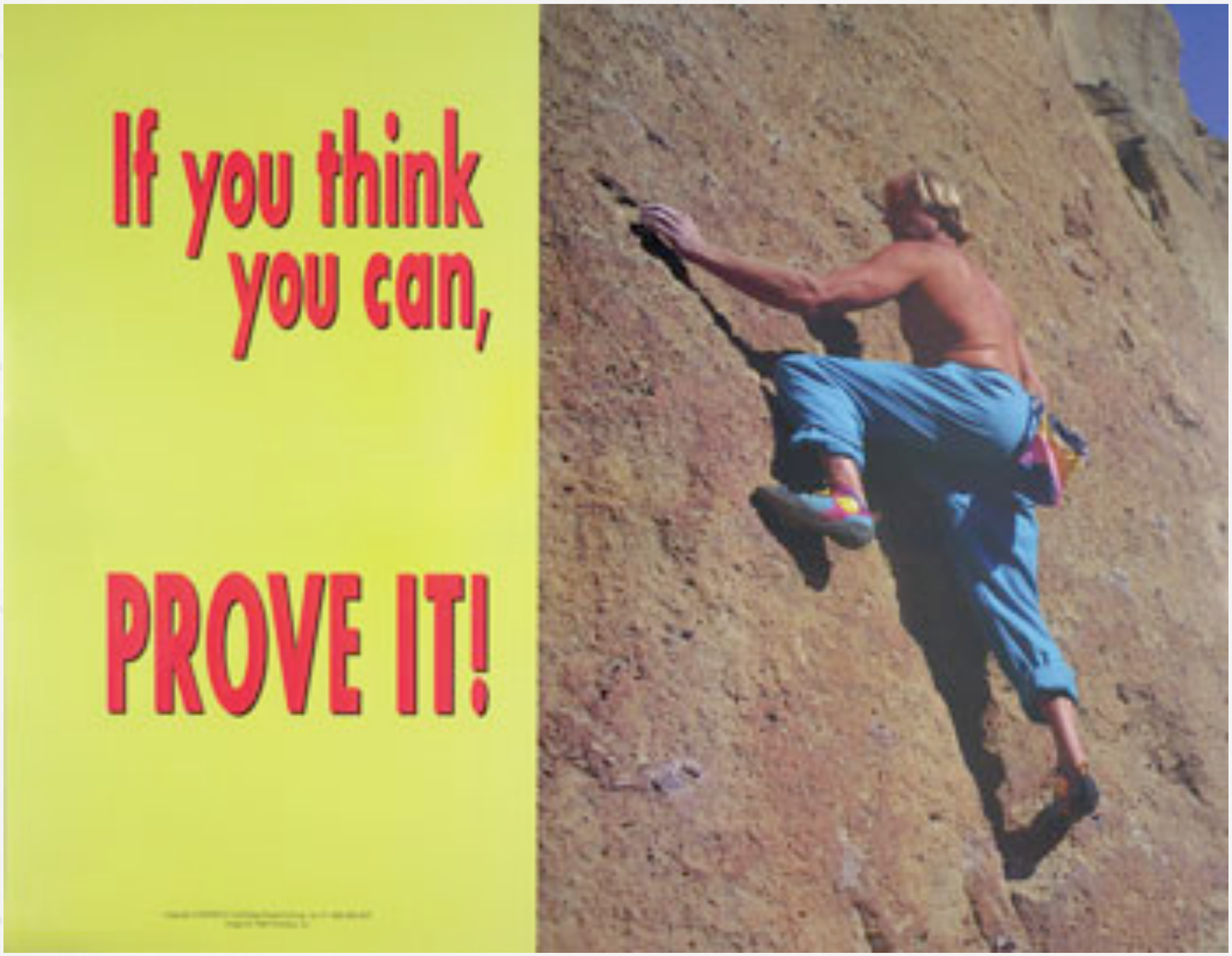
...will change your life



Test-Driven Development

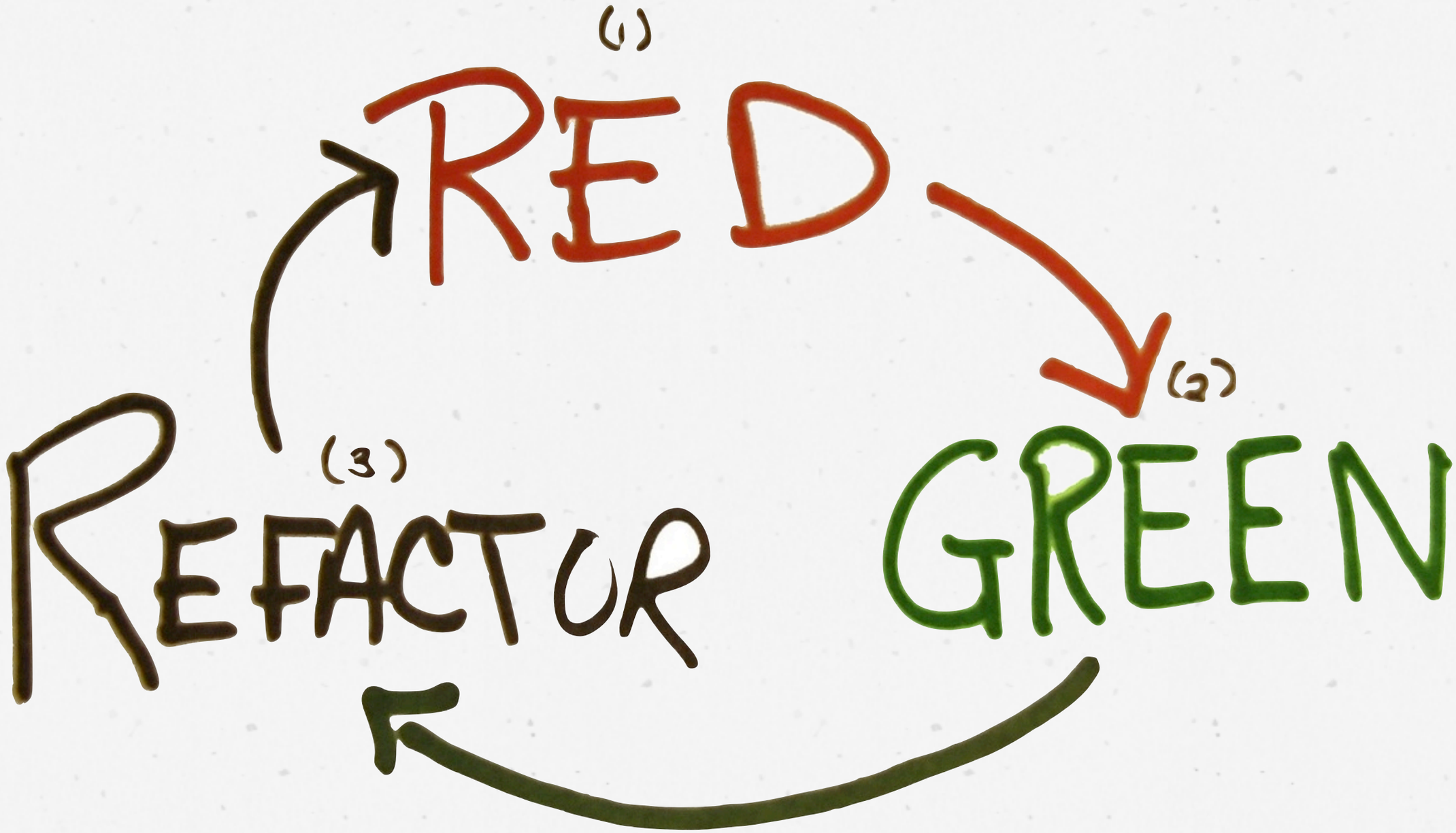
...will change your life

What good is a test?



Verification & Confidence

Documentation of Intention



A New Way of Thinking

Team

Pair

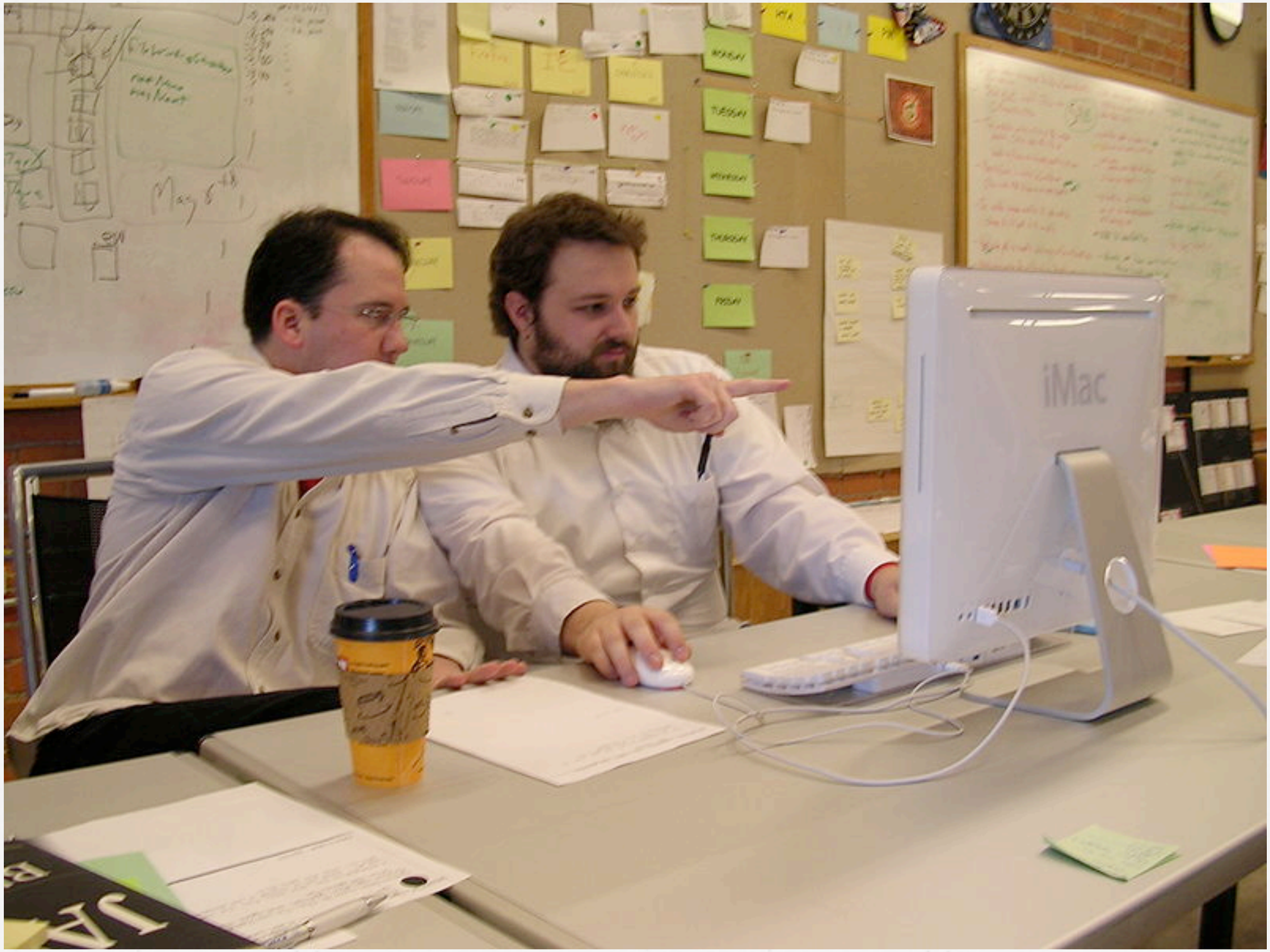
Code

Tools

Tech Stack

Pair Programming

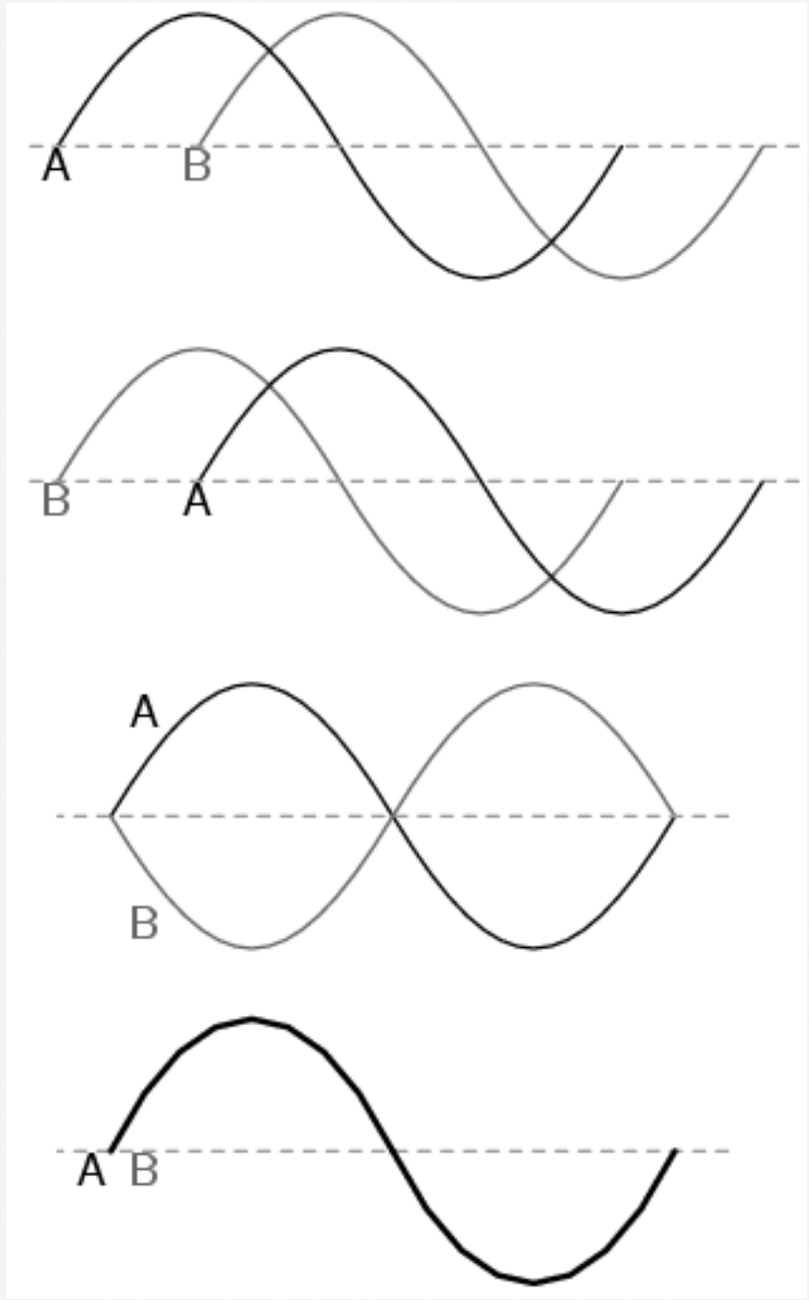
What is pair programming?


iMac

What good is a pair?

Constant Code Review

Construction as Conversation



Partner in Crime

How we pair

Strategize together

Write a failing test

ping

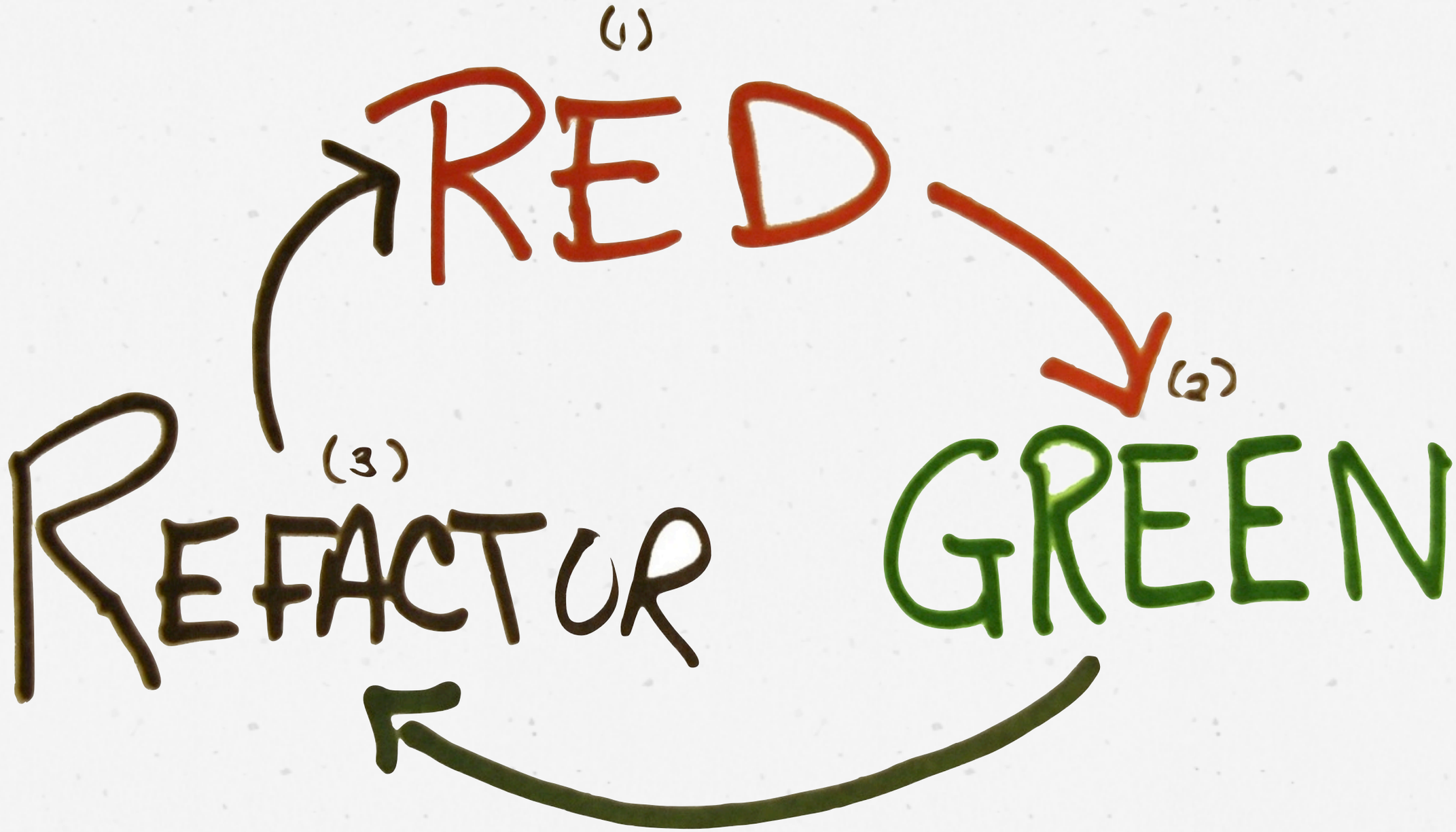
Make test pass

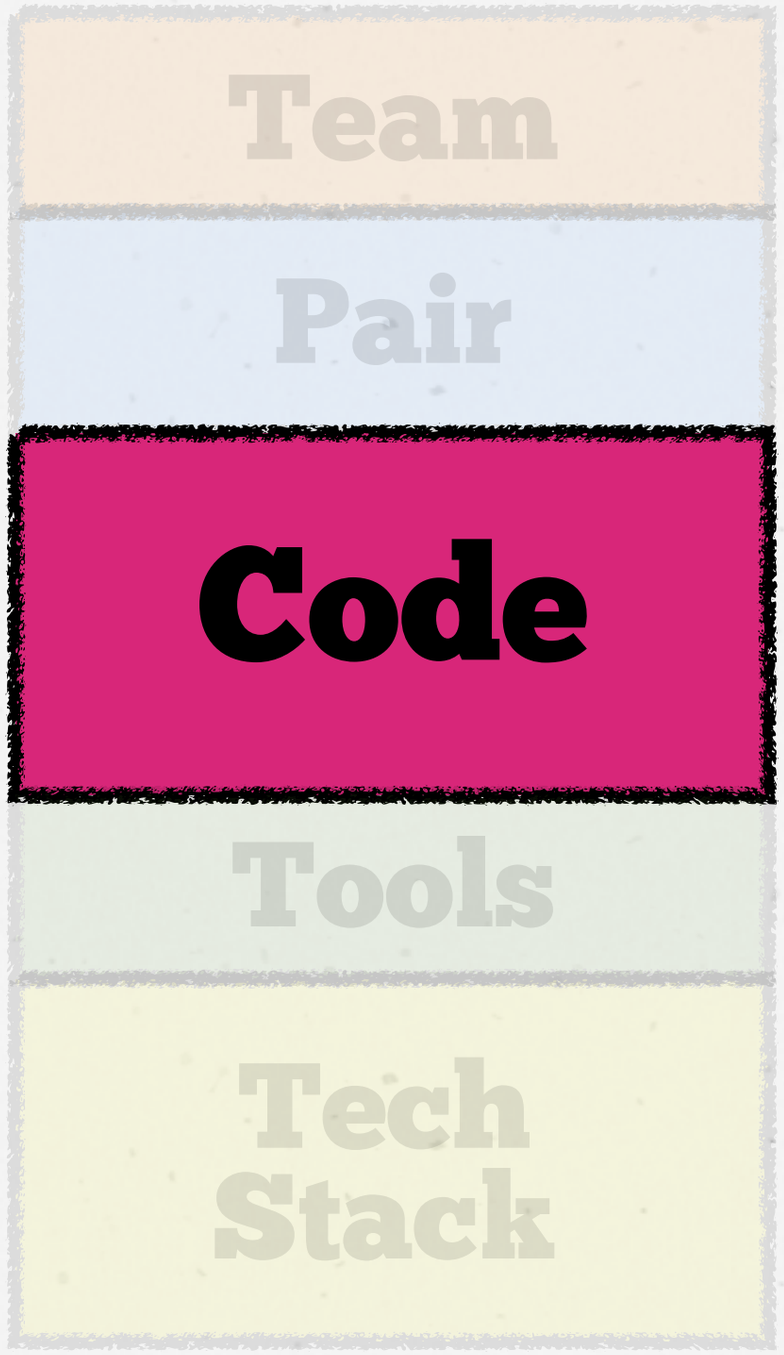
Write another failing test

pong

Make test pass

...and so on

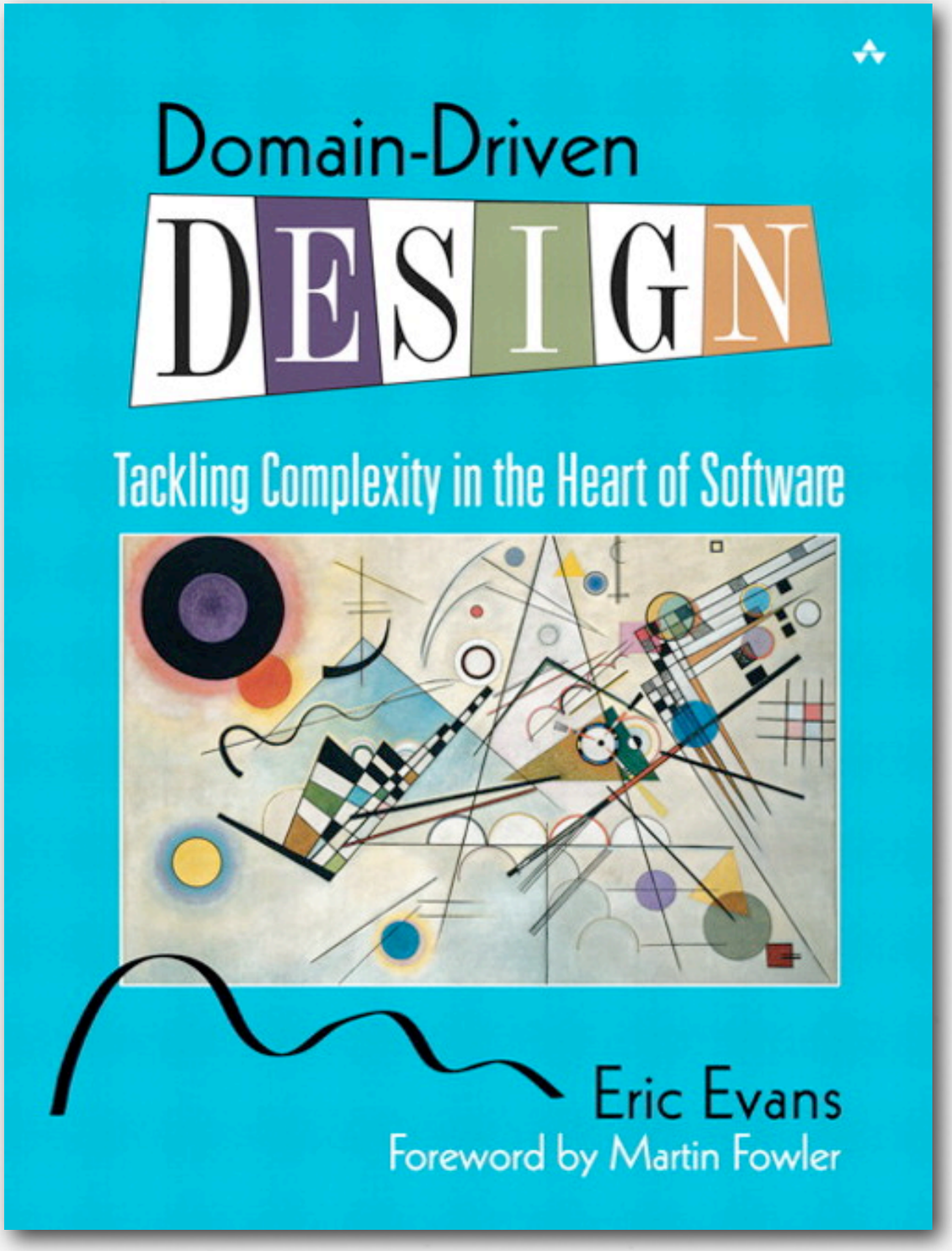

(1)
RED
(2)
GREEN
(3)
REFACTOR

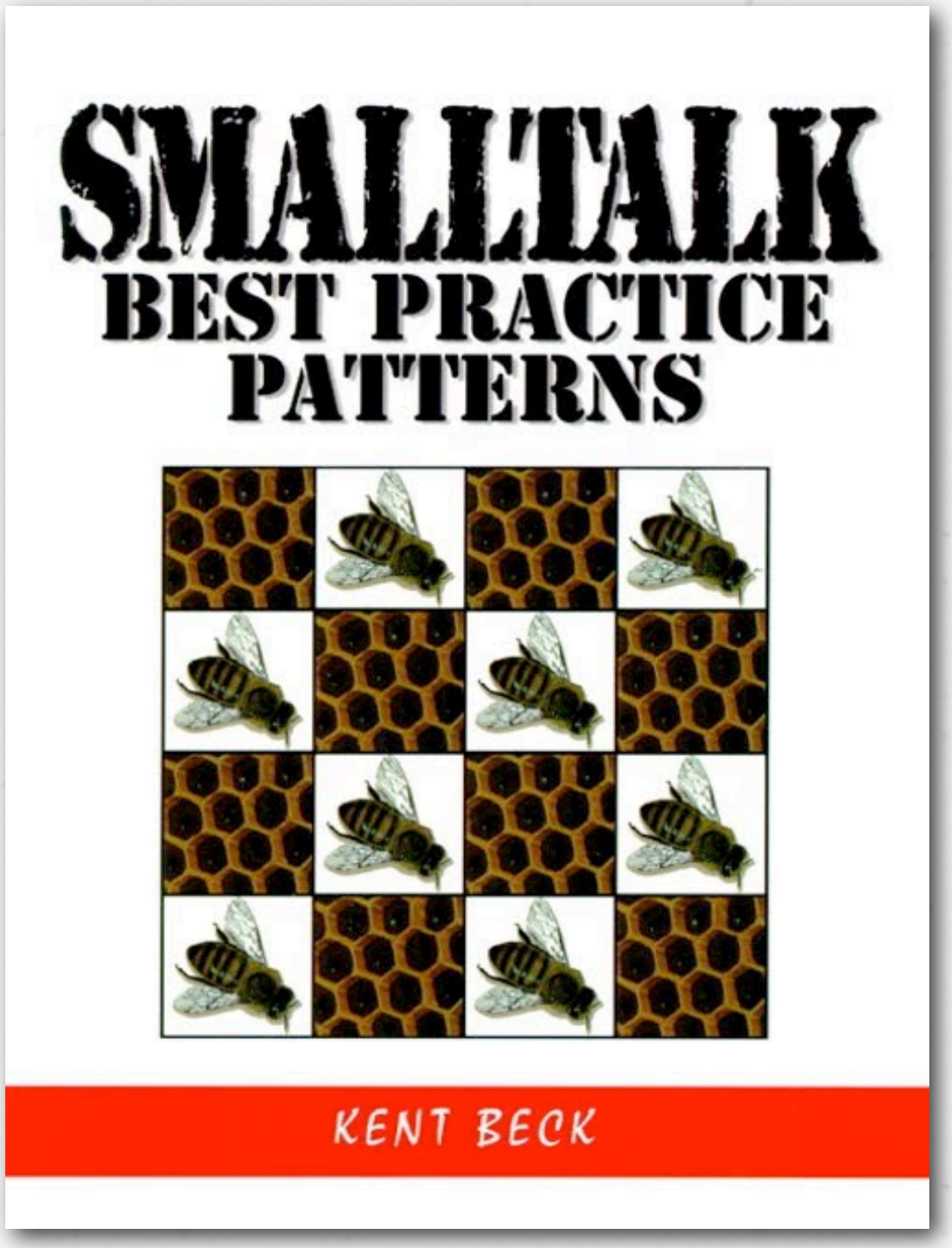


How we code

Domain over everything

domain on my mind





Names for things we do

Polyglot?!

Polyglot?!

Polyglot!

Clean APIs

```java
import java.math.BigDecimal;
import com.braintreegateway.*;

public class BraintreeExample {
    public static void main(String[] args) {
        BraintreeGateway gateway = new BraintreeGateway(
            Environment.SANDBOX,
            "the_merchant_id",
            "the_public_key",
            "the_private_key"
        );
        TransactionRequest request = new TransactionRequest().
            amount(new BigDecimal("1000.00")).
            creditCard().
                number("4111111111111111").
                expirationDate("05/2009").
                done();
        Result<Transaction> result = gateway.transaction().sale(request);
        if (result.isSuccess()) {
            Transaction transaction = result.getTarget();
            System.out.println("Success!: " + transaction.getId());
        } else if (result.getTransaction() != null) {
            System.out.println("Message: " + result.getMessage());
            Transaction transaction = result.getTransaction();
            System.out.println("Error processing transaction:");
            System.out.println("  Status: " + transaction.getStatus());
            System.out.println("  Code: " + transaction.getProcessorResponseCode());
            System.out.println("  Text: " + transaction.getProcessorResponseText());
        } else {
            System.out.println("Message: " + result.getMessage());
            for (ValidationError error : result.getErrors().getAllDeepValidationErrors()) {
                System.out.println("Attribute: " + error.getAttribute());
                System.out.println("  Code: " + error.getCode());
                System.out.println("  Message: " + error.getMessage());
            }
        }
    }
}
```


```
using System;
using Braintree;
namespace BraintreeExample
{
    class Program
    {
        static void Main(string[] args)
        {
            var gateway = new BraintreeGateway
            {
                Environment = Braintree.Environment.SANDBOX,
                MerchantId = "the_merchant_id",
                PublicKey = "a_public_key",
                PrivateKey = "a_private_key"
            };
            var request = new TransactionRequest
            {
                Amount = 1000M,
                CreditCard = new TransactionCreditCardRequest
                {
                    Number = "4111111111111111",
                    ExpirationDate = "05/2012"
                }
            };
            Result<Transaction> result = gateway.Transaction.Sale(request);

            if (result.IsSuccess())
            {
                Transaction transaction = result.Target;
                Console.WriteLine("Success!: " + transaction.Id);
            }
            else if (result.Transaction != null)
            {
                Console.WriteLine("Message: " + result.Message);
                Transaction transaction = result.Transaction;
                Console.WriteLine("Error processing transaction:");
                Console.WriteLine("  Status: " + transaction.Status);
                Console.WriteLine("  Code: " + transaction.ProcessorResponseCode);
                Console.WriteLine("  Text: " + transaction.ProcessorResponseText);
```


```
};
var request = new TransactionRequest
{
    Amount = 1000M,
    CreditCard = new TransactionCreditCardRequest
    {
        Number = "4111111111111111",
        ExpirationDate = "05/2012"
    }
};
Result<Transaction> result = gateway.Transaction.Sale(request);

if (result.IsSuccess())
{
    Transaction transaction = result.Target;
    Console.WriteLine("Success!: " + transaction.Id);
}
else if (result.Transaction != null)
{
    Console.WriteLine("Message: " + result.Message);
    Transaction transaction = result.Transaction;
    Console.WriteLine("Error processing transaction:");
    Console.WriteLine("  Status: " + transaction.Status);
    Console.WriteLine("  Code: " + transaction.ProcessorResponseCode);
    Console.WriteLine("  Text: " + transaction.ProcessorResponseText);
}
else
{
    Console.WriteLine("Message: " + result.Message);
    foreach (ValidationError error in result.Errors.DeepAll())
    {
        Console.WriteLine("Attribute: " + error.Attribute);
        Console.WriteLine("  Code: " + error.Code);
        Console.WriteLine("  Message: " + error.Message);
    }
}
    }
}
```

```
var sys = require('sys'),
    braintree = require('braintree');

var gateway = braintree.connect({
  environment: braintree.Environment.Sandbox,
  merchantId: 'your_merchant_id',
  publicKey: 'your_public_key',
  privateKey: 'your_private_key'
});

gateway.transaction.sale({
  amount: '5.00',
  creditCard: {
    number: '5105105105105100',
    expirationDate: '05/12'
  }
}, function (err, result) {
  if (err) throw err;

  if (result.success) {
    sys.puts('Transaction ID: ' + result.transaction.id);
  } else {
    sys.puts(result.message);
  }
});
```


```perl
use Net::Braintree;

my $config = Net::Braintree->configuration;
$config->environment("sandbox");
$config->merchant_id("your_merchant_id");
$config->public_key("your_public_key");
$config->private_key("your_private_key");

my $result = Net::Braintree::Transaction->sale({
  amount => "1000.00",
  credit_card => {
    number => "5105510551055100",
    expiration_date => "05/12"
  }
});

if ($result->is_success) {
  printf "Transaction ID: %s", $result->transaction->id;
  # status will be authorized or submitted_for_settlement
  printf "Transaction Status: %s", $result->transaction->status;
} else {
  printf "Message: %s", $result->message;
  if !defined($result->transaction) {
    # validation errors prevented transaction from being created
    print $result->errors;
  } else {
    printf "Transaction ID: %s", $result->transaction->id;
    # status will be processor_declined, gateway_rejected, or failed
    printf "Transaction Status: %s", $result->transaction->status;
  }
}
```

```php
require_once 'PATH_TO_BRAINTREE/lib/Braintree.php';

Braintree_Configuration::environment('sandbox');
Braintree_Configuration::merchantId('your_merchant_id');
Braintree_Configuration::publicKey('your_public_key');
Braintree_Configuration::privateKey('your_private_key');

$result = Braintree_Transaction::sale(array(
    'amount' => '1000.00',
    'creditCard' => array(
        'number' => '5105105105105100',
        'expirationDate' => '05/12'
    )
));

if ($result->success) {
    print_r("success!: " . $result->transaction->id);
} else if ($result->transaction) {
    print_r("Error processing transaction:");
    print_r("\n  message: " . $result->message);
    print_r("\n  code: " . $result->transaction->processorResponseCode);
    print_r("\n  text: " . $result->transaction->processorResponseText);
} else {
    print_r("Message: " . $result->message);
    print_r("\nValidation errors: \n");
    print_r($result->errors->deepAll());
}
```


```python
import braintree

braintree.Configuration.configure(
    braintree.Environment.Sandbox,
    "the_merchant_id",
    "the_public_key",
    "the_private_key"
)

result = braintree.Transaction.sale({
    "amount": "1000.00",
    "credit_card": {
        "number": "4111111111111111",
        "expiration_date": "05/2012"
    }
})

if result.is_success:
    print "success!: " + result.transaction.id
elif result.transaction:
    print "Error processing transaction:"
    print "  message: " + result.message
    print "  code:    " + result.transaction.processor_response_code
    print "  text:    " + result.transaction.processor_response_text
else:
    print "message: " + result.message
    for error in result.errors.deep_errors:
        print "attribute: " + error.attribute
        print "  code: " + error.code
        print "  message: " + error.message
```


```ruby
require "rubygems"
require "braintree"

Braintree::Configuration.environment = :sandbox
Braintree::Configuration.merchant_id = "your_merchant_id"
Braintree::Configuration.public_key = "your_public_key"
Braintree::Configuration.private_key = "your_private_key"

result = Braintree::Transaction.sale(
  :amount => "1000.00",
  :credit_card => {
    :number => "5105105105105100",
    :expiration_date => "05/12"
  }
)

if result.success?
  puts "Transaction ID: #{result.transaction.id}"
  # status will be authorized or submitted_for_settlement
  puts "Transaction Status: #{result.transaction.status}"
else
  puts "Message: #{result.message}"
  if result.transaction.nil?
    # validation errors prevented transaction from being created
    p result.errors
  else
    puts "Transaction ID: #{result.transaction.id}"
    # status will be processor_declined, gateway_rejected, or failed
    puts "Transaction Status: #{result.transaction.status}"
  end
end
```

Good Polyglot =

Pragmatic Polyglot

Team

Pair

Code

Tools

Tech Stack

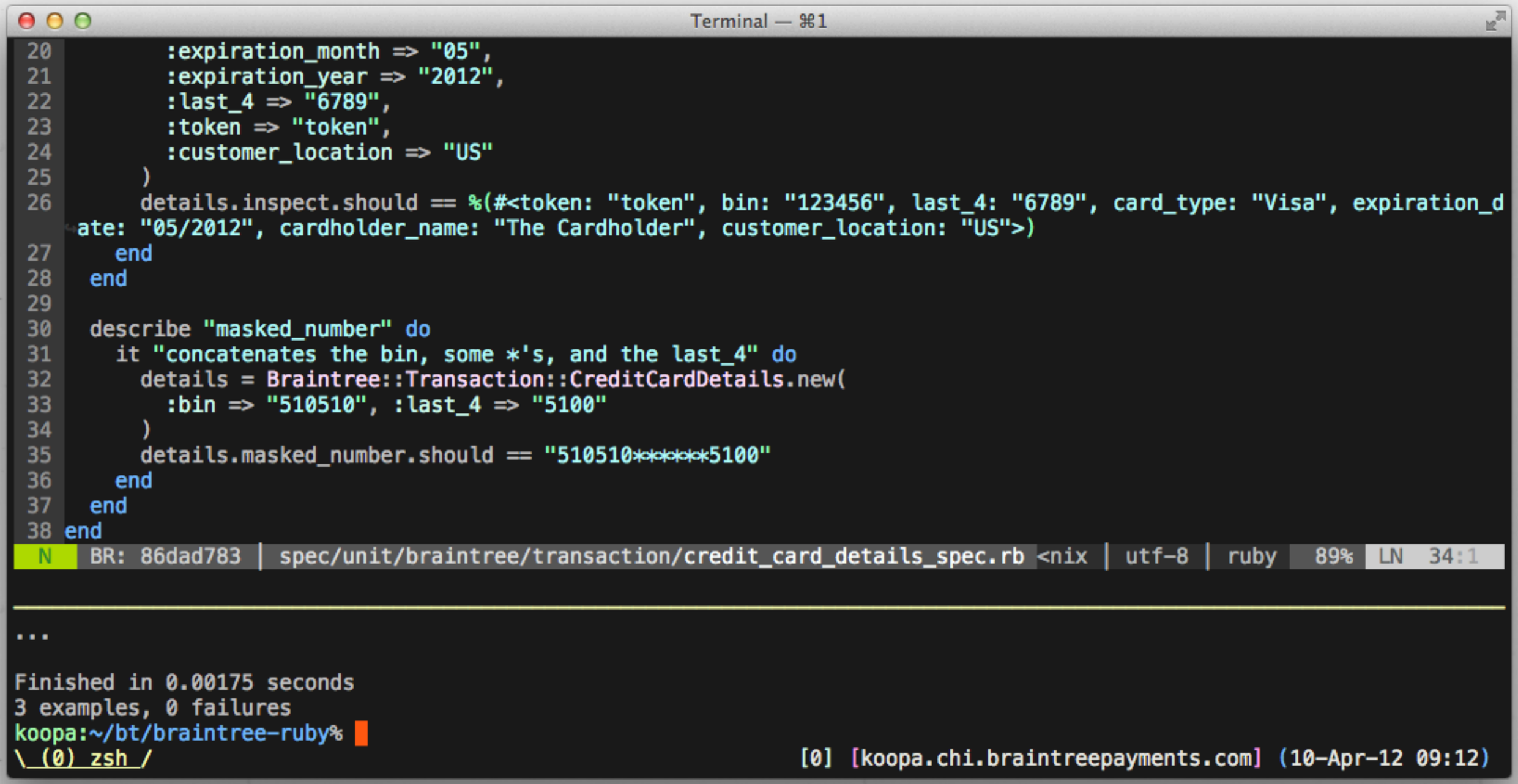
Stuff we use

Just use git

For godssake use git

Also use GitHub

Elevate your language


Terminal — ⌘1
20 :expiration_month => "05",
21 :expiration_year => "2012",
22 :last_4 => "6789",
23 :token => "token",
24 :customer_location => "US"
25)
26 details.inspect.should == %(#<token: "token", bin: "123456", last_4: "6789", card_type: "Visa", expiration_d
ate: "05/2012", cardholder_name: "The Cardholder", customer_location: "US">)
27 end
28 end
29
30 describe "masked_number" do
31 it "concatenates the bin, some *'s, and the last_4" do
32 details = Braintree::Transaction::CreditCardDetails.new(
33 :bin => "510510", :last_4 => "5100"
34)
35 details.masked_number.should == "510510******5100"
36 end
37 end
38 end
N BR: 86dad783 | spec/unit/braintree/transaction/credit_card_details_spec.rb <nix | utf-8 | ruby 89% LN 34:1
...
Finished in 0.00175 seconds
3 examples, 0 failures
koopa:~/bt/braintree-ruby%
\ (0) zsh /
[0] [koopa.chi.braintreepayments.com] (10-Apr-12 09:12)

vim + tmux = <3

unix tools

```
...
Apr 30 18:11:47 "POST /merchants/tcwhwpq75vyf7pp9/transparent_redirect_requests/w8xphcwhxnb8hh54/confi
Apr 30 18:11:48 "PUT /merchants/p5rnfb7stsyc8zyh/transactions/5m2xmm/submit_for_settlement HTTP/1.1" 2
Apr 30 18:11:48 "GET /merchants/2b9y47kwmx6m4rv2/customers/x HTTP/1.1" 404
Apr 30 18:11:48 "PUT /merchants/tcwhwpq75vyf7pp9/transactions/hbzmqg/submit_for_settlement HTTP/1.1" 2
Apr 30 18:11:49 "POST /merchants/tw6fybj4vfy34z55/customers HTTP/1.1" 201
Apr 30 18:11:49 "POST /merchants/tw6fybj4vfy34z55/transparent_redirect_requests/8cpv6fmkggsb28rf/confi
Apr 30 18:11:49 "GET /merchants/pssrj367r8h4m3n8/subscriptions/1234873727 HTTP/1.1" 200
Apr 30 18:11:50 "GET /merchants/pssrj367r8h4m3n8/subscriptions/1234873727 HTTP/1.1" 200
Apr 30 18:11:51 "POST /merchants/tw6fybj4vfy34z55/transactions HTTP/1.1" 201
Apr 30 18:11:51 "POST /merchants/tw6fybj4vfy34z55/customers HTTP/1.1" 201
Apr 30 18:11:52 "GET /merchants/mph2sx3hnprn7m37/customers/4f9ed3ea83ac6721ae00000d HTTP/1.1" 200
Apr 30 18:11:52 "GET /merchants/pssrj367r8h4m3n8/customers/2116426642 HTTP/1.1" 200
Apr 30 18:11:52 "POST /merchants/tw6fybj4vfy34z55/customers HTTP/1.1" 201
Apr 30 18:11:52 "GET /merchants/grc7zw68p9fybcrx/plans HTTP/1.1" 200
Apr 30 18:11:52 "GET /merchants/fdcpggzf8kgjvxnz/customers/1 HTTP/1.1" 200
Apr 30 18:11:52 "GET /merchants/pssrj367r8h4m3n8/customers/2116426642 HTTP/1.1" 200
Apr 30 18:11:53 "GET /merchants/grc7zw68p9fybcrx/plans HTTP/1.1" 200
Apr 30 18:11:54 "POST /merchants/6kdffyjmf2sgm653/customers/advanced_search_ids HTTP/1.1" 200
Apr 30 18:11:54 "GET /merchants/pssrj367r8h4m3n8/subscriptions/1285787219 HTTP/1.1" 200
Apr 30 18:11:55 "POST /merchants/tw6fybj4vfy34z55/customers HTTP/1.1" 201
Apr 30 18:11:55 "GET /merchants/pssrj367r8h4m3n8/subscriptions/1285787219 HTTP/1.1" 200
Apr 30 18:11:56 "GET /merchants/tw6fybj4vfy34z55/customers/3750786 HTTP/1.1" 200
Apr 30 18:11:56 "GET /merchants/pssrj367r8h4m3n8/customers/508837329 HTTP/1.1" 200
...
```

cat apache.log

```
...
Apr 29 06:46:52 "POST /merchants/tcwhwpq75vyf7pp9/transparent_redirect_requests HTTP/1.1" 303
Apr 29 06:46:54 "POST /merchants/tcwhwpq75vyf7pp9/transparent_redirect_requests/59vk7g5gvd8f85mk/confi
Apr 29 06:46:54 "POST /merchants/6kdffyjmf2sgm653/customers/advanced_search_ids HTTP/1.1" 200
Apr 29 06:46:55 "PUT /merchants/tcwhwpq75vyf7pp9/transactions/9mtknr/submit_for_settlement HTTP/1.1" 2
Apr 29 06:47:01 "POST /merchants/78vfmgx8ghw425mf/transactions HTTP/1.1" 422
Apr 29 06:47:02 "GET /merchants/f2tdk79y64gmj2tx/transactions/jb6tgr HTTP/1.1" 404
Apr 29 06:47:03 "GET /merchants/f2tdk79y64gmj2tx/transactions/6gz37b HTTP/1.1" 404
Apr 29 06:47:03 "POST /merchants/sfs23ncfk37xmkf4/payment_methods/all/expiring_ids?start=042012&end=05
Apr 29 06:47:05 "POST /merchants/sfs23ncfk37xmkf4/payment_methods/all/expiring?start=042012&end=052012
Apr 29 06:47:05 "POST /merchants/x4y2tbn3zv5kxmwn/transparent_redirect_requests HTTP/1.1" 303
Apr 29 06:47:05 "POST /merchants/x4y2tbn3zv5kxmwn/transparent_redirect_requests/zr4h9x7y4xf33d7h/confi
Apr 29 06:47:05 "GET /merchants/x4y2tbn3zv5kxmwn/customers/213 HTTP/1.1" 404
Apr 29 06:47:05 "GET /merchants/f2tdk79y64gmj2tx/transactions/8n539m HTTP/1.1" 404
Apr 29 06:47:06 "GET /merchants/4x5jxfvh25cx4mjw/plans HTTP/1.1" 200
Apr 29 06:47:07 "GET /merchants/56xv4qhtvwmrynxx/customers/33410 HTTP/1.1" 200
Apr 29 06:47:07 "GET /merchants/4x5jxfvh25cx4mjw/plans HTTP/1.1" 200
Apr 29 06:47:13 "GET /merchants/6r9z9pr6hb48zyy8/customers/INVALIDID HTTP/1.1" 404
Apr 29 06:47:14 "POST /merchants/6kdffyjmf2sgm653/customers/advanced_search_ids HTTP/1.1" 200
...
```

```
cat apache.log | grep 'Apr 29'
```

```
Apr 29 07:00:00 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/c6gqh6/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:00 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/k6hr7r/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:00 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/44pz62/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:01 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/594k3r/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:01 "POST /merchants/78vfmgx8ghw425mf/transactions HTTP/1.1" 422
Apr 29 07:00:01 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/4y2ss6/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:02 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/gxtjrm/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:02 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:02 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/g934m2/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:02 "POST /merchants/zcf46s3zj8myyrqd/payment_methods/all/expiring_ids?start=042012&end=06
Apr 29 07:00:02 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:02 "GET /merchants/6r9z9pr6hb48zyy8/transactions/fkbx2m HTTP/1.1" 200
Apr 29 07:00:02 "PUT /merchants/bqdcwtwf7j5fzjnn/transactions/d3nhpw/submit_for_settlement HTTP/1.1" 4
Apr 29 07:00:02 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:03 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:03 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:03 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:03 "POST /merchants/cn3nt6g3dyf8cgmk/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:03 "POST /merchants/6qgdwz75nd8tc9cv/transactions/advanced_search_ids HTTP/1.1" 200
Apr 29 07:00:03 "POST /merchants/snw7jr55grjvj95m/payment_methods HTTP/1.1" 201
...
```

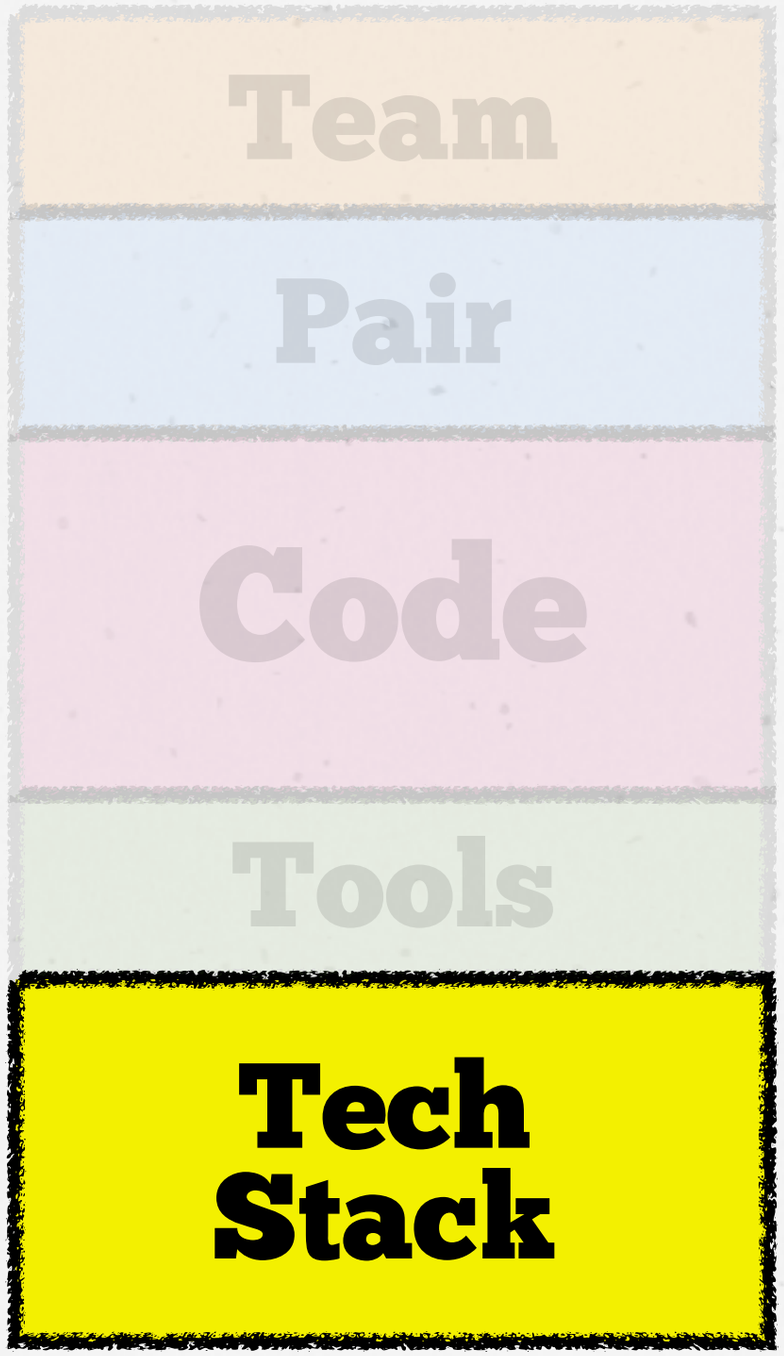
```
cat apache.log | grep 'Apr 29 0[789]'
```

```
422
422
422
422
422
422
422
200
422
200
200
200
422
200
200
200
200
200
201
...
```

```
cat apache.log | grep 'Apr 28 0[345]' \
    | awk '{ print $11 }'
```

```
3768 200
 410 201
  42 302
  54 303
  22 304
  56 401
3035 404
 423 422
```

```
cat apache.log | grep 'Apr 28 0[345]' |
awk '{ print $11 }' | sort -n | uniq -c
```



Highly Available Architecture

Stop, wait!

"Full Stack" Developer

To the chalkboard!