A Ruby on Rails app interacts with two services and a database.
Pete Hodgson
@ph1, Glorified Typist

ThoughtWorks
315 Montgomery Street, 16th Floor | San Francisco, California | 94104
T: +1 415 869 3000  C: +1 707 206 8897
E: phodgson@thoughtworks.com
ACT ONE

CO-DEPENDENCE and INDEPENDENCE
SELECTION OF DEALS

ARRAY OF ENTICING PRODUCT CATEGORIES
Rails app

Product service

deals service
CONWAY'S LAW
HOW PEOPLE WORK TOGETHER AFFECTS SYSTEM ARCHITECTURE
SYSTEM ARCHITECTURE AFFECTS HOW PEOPLE WORK TOGETHER
BLUE TEAM: product service

GREEN TEAM: deals service

RED TEAM: rails app
WHO'S BUG IS IT ANYWAY?
BOUNDED INTEGRATION TESTS

rails app

fake service

fake service

@ph1
IN PROCESS

vcr

webmock

OUT-OF-PROCESS

mimic

moco

stubby

montebank
CONTRACT TESTS
rails
app

product service

deals service
plain old rspec
(or minitest, or whatever)
pacto
pact
UNIT TESTS

FUNCTIONAL TESTS

BOUNDED INTEGRATION TESTS

END-TO-END TESTS

SERVICE A
CONTRACT TESTS

SERVICE B
CONTRACT TESTS

SERVICE C
CONTRACT TESTS

SERVICE D
CONTRACT TESTS

FUNCTIONAL TESTS

SERVICE A
CONTRACT TESTS

SERVICE B
CONTRACT TESTS

SERVICE C
CONTRACT TESTS

SERVICE D
CONTRACT TESTS
UNIT TESTS

FUNCTIONAL TESTS

BOUNDED INTEGRATION TESTS

END-TO-END TESTS

SERVICE A CONTRACT TESTS

SERVICE B CONTRACT TESTS

SERVICE C CONTRACT TESTS

SERVICE D CONTRACT TESTS
ACT TWO

SERVICE GATEWAYS
use Rack::Sendfile
use ActionDispatch::Static
use Rack::Lock
use ActiveSupport::Cache::Strategy::LocalCache::Middleware
use Rack::Runtime
use Rack::MethodOverride
use ActionDispatch::RequestId
use Rails::Rack::Logger
use ActionDispatch::ShowExceptions
use ActionDispatch::DebugExceptions
use ActionDispatch::RemoteIp
use ActionDispatch::Reloader
use ActionDispatch::Callbacks
use ActiveRecord::Migration::CheckPending
use ActiveRecord::ConnectionAdapters::ConnectionManagement
use ActiveRecord::QueryCache
use ActionDispatch::Cookies
use ActionDispatch::Session::CookieStore
use ActionDispatch::Flash
use ActionDispatch::ParamsParser
use Rack::Head
use Rack::ConditionalGet
use Rack::ETag
run MyApp::Application.routes
CACHING
AUTHENTICATION
SERIALIZATION
HTTP CLIENT
class BikeStationGateway
  STATIONS_PATH = '/stations/json'

  def initialize( base_url = 'http://bayareabikeshare.com' )
    @conn = Faraday.new(:url => base_url ) do |faraday|
      faraday.use :instrumentation
      faraday.response :json
      faraday.request :encode_json
      faraday.response :follow_redirects
      faraday.response :logger
      faraday.adapter  Faraday.default_adapter
    end
  end

  def stations
    response = @conn.get(STATIONS_PATH)
    # ... parse response.body into a list of stations
  end
end
class BikeStationGateway
  STATIONS_PATH = '/stations/json'

  def initialize( base_url = 'http://bayareabikeshare.com' )
    @conn = Faraday.new(:url => base_url ) do |faraday|
      faraday.use :instrumentation
      faraday.response :json
      faraday.request :encode_json
      faraday.response :follow_redirects
      faraday.response :logger
      faraday.adapter  Faraday.default_adapter
    end
  end

  def stations
    response = @conn.get(STATIONS_PATH)
    # ... parse response.body into a list of stations
  end
end
service gateways
hexagonal architecture
{  "JSON": "parsing" }

<xml>
  <parsing/>
</xml>
hashie    lazy_doc
embedded_doc
representable
cache the crap out of this!
cache the crap out of this too!
CACHING
AUTHENTICATION
HTTP CLIENT
SERIALIZATION
class BikeStationGateway
  STATIONS_PATH = '/stations/json'

  def initialize( base_url = 'http://bayareabikeshare.com' )
    @conn = Faraday.new(:url => base_url ) do |faraday|
      faraday.use :instrumentation
      faraday.response :json
      faraday.request :encode_json
      faraday.response :follow_redirects
      faraday.response :logger

      faraday.use
        FaradayMiddleware::RackCompatible,
        Rack::Cache::Context,
        :metastore => "file:#{cache_dir}/faraday/meta",
        :entitystore => "file:#{cache_dir}/faraday/body",
        :ignore_headers => %w[Set-Cookie X-Content-Digest]

      faraday.adapter Faraday.default_adapter
    end
  end
end
on the web

OF the web
EPILOGUE

SOME PRINCIPLES
CONWAY'S LAW
POSTEL’S LAW
HEXAGONAL ARCHITECTURE
EMBRACE THE WEB
THANKS!

Pete Hodgson
@ph1
phodgson@thoughtworks.com