Pacts to the rescue!

Making your microservices play nicely together
with Consumer Driven Contracts

Beth Skurrie
@bethesque / @pact_up
bskurrie@dius.com.au

“Integrated tests are a scam.
A self replicating virus
that threatens
the very health of your codebase,
your sanity,
and I'm not exaggerating when I say,
your life.”

- JB Rainsberger

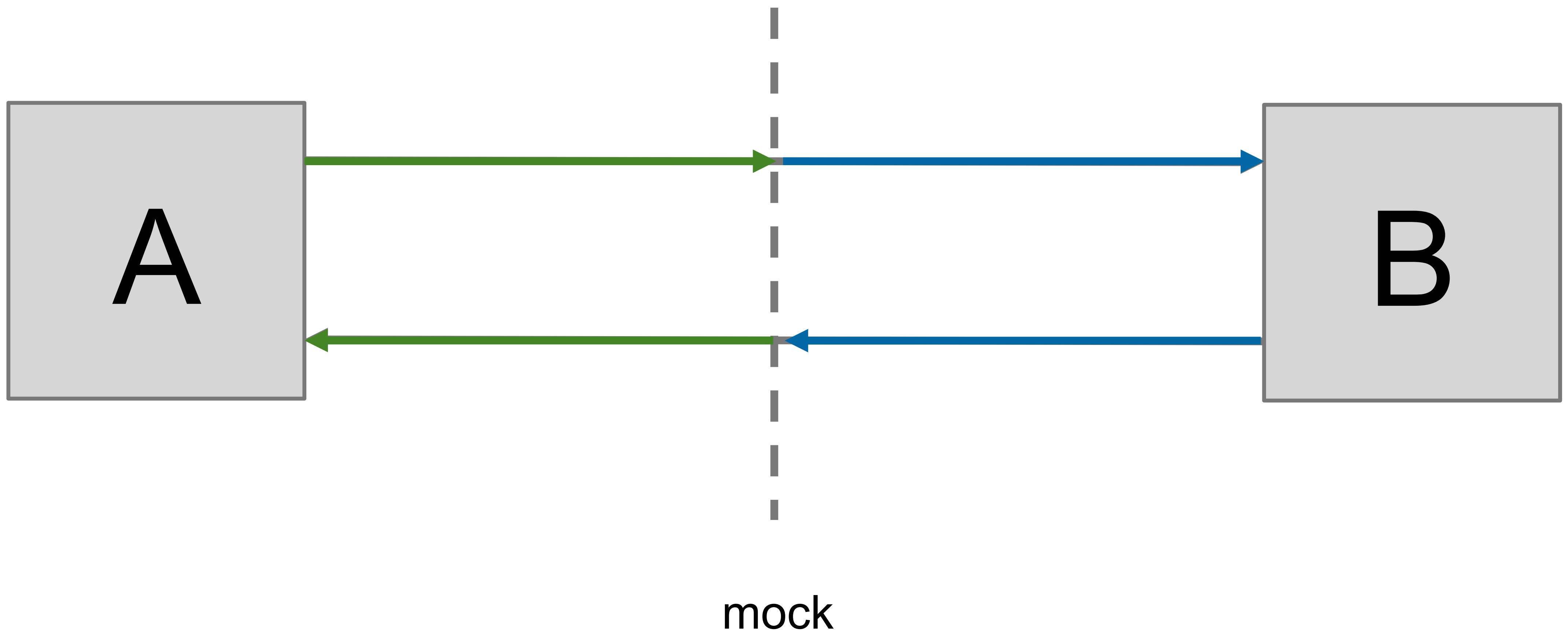
Integrated Tests: The Problems

- Slow
- Easy to break
- Hard to fix
- Scales combinatorially

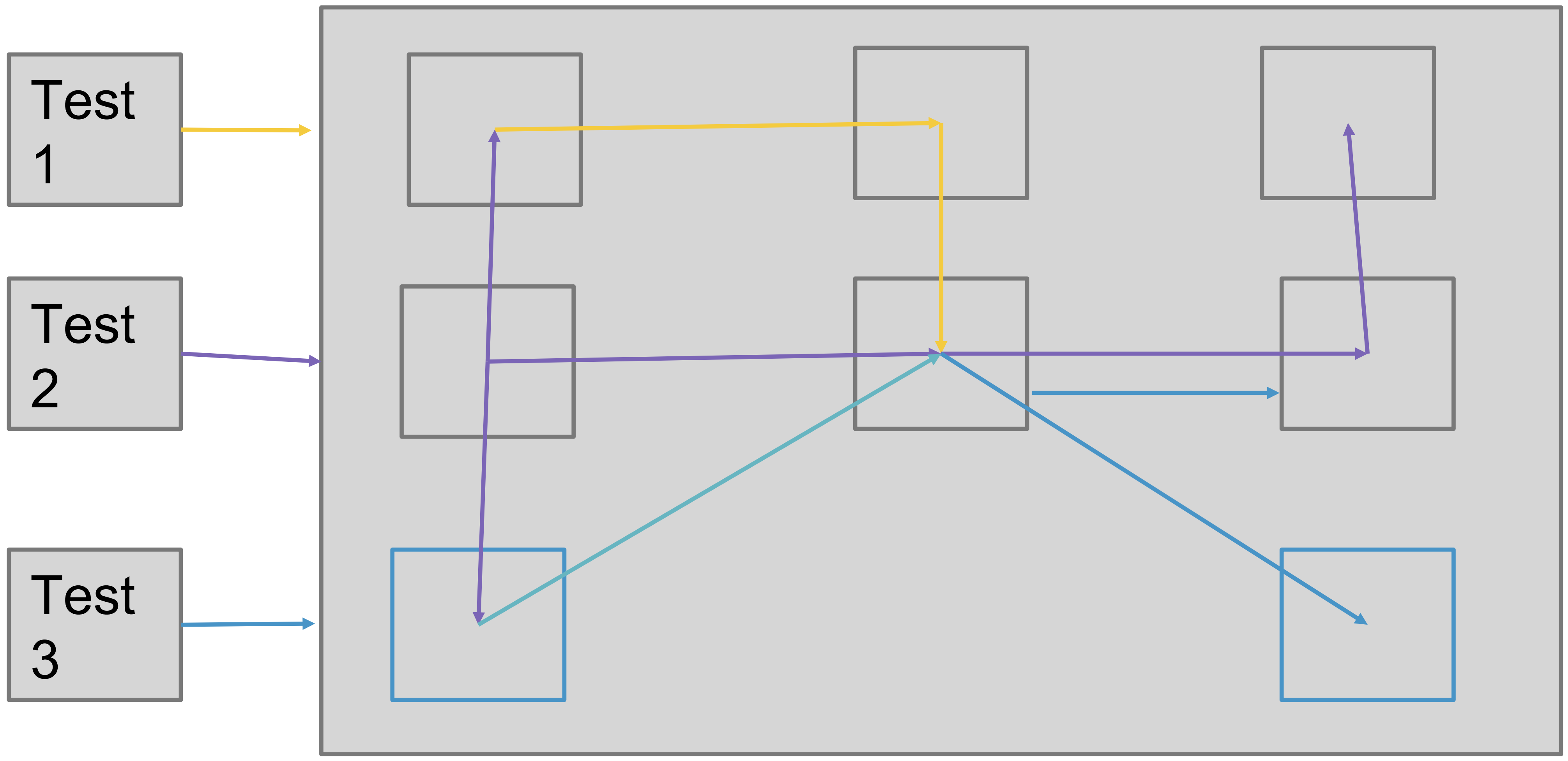
3 classes, 4 code paths each

4 * 4 * 4 = 64 tests

Test Symmetry… how it works



Integrated tests



Test Symmetry: The Problem

- Lack of automated tools
- Relies on developer eternal vigilance
- Does not scale

Posed to Dius

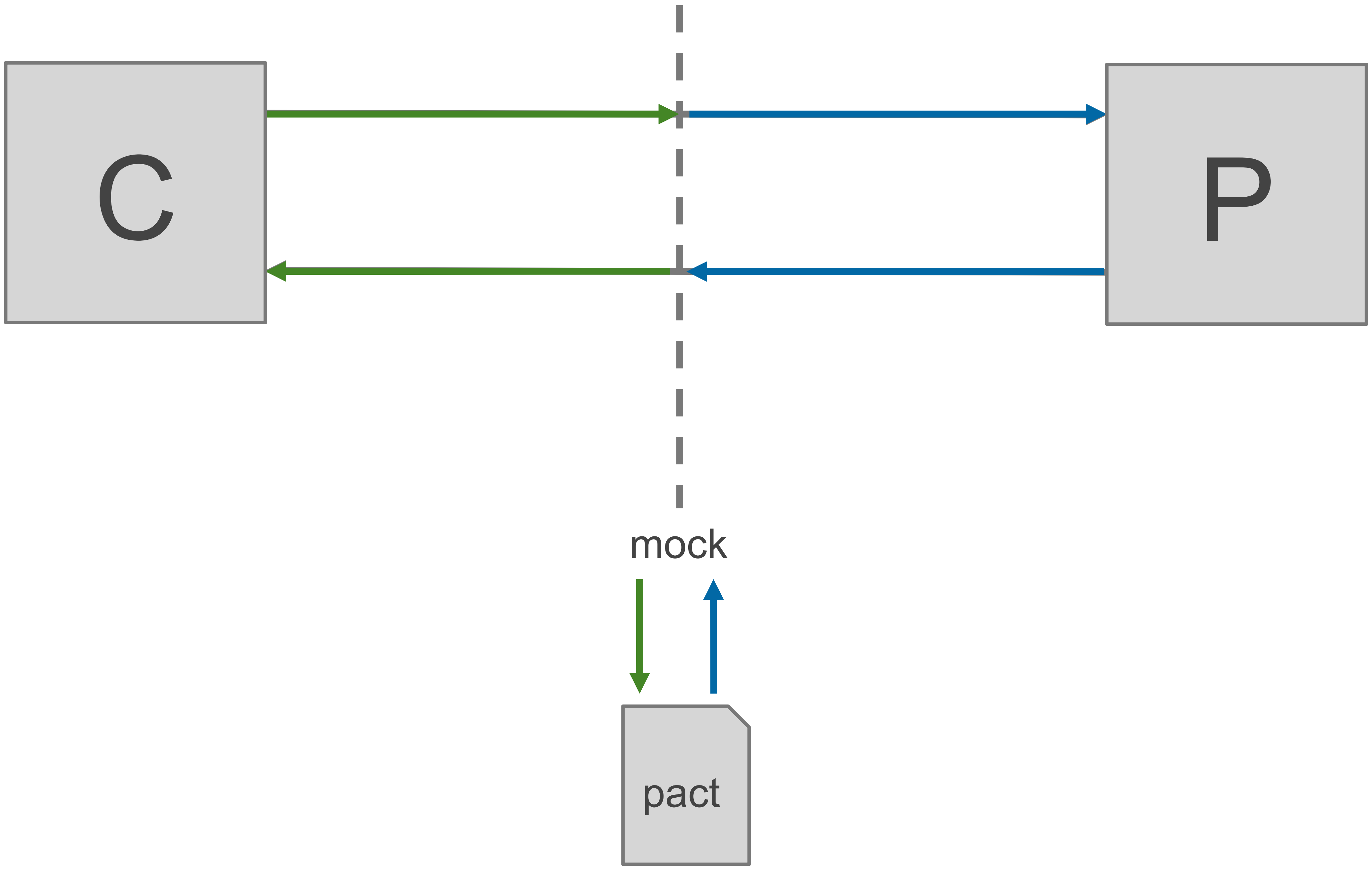
Do we have the technology to achieve test symmetry?

System Integration Tests: The WORST

- Slower
- Easier to break
- Harder to fix
- Scales BADLY
- Lots of set up
- Flakey
- Extra infrastructure

Pact to the rescue!!!

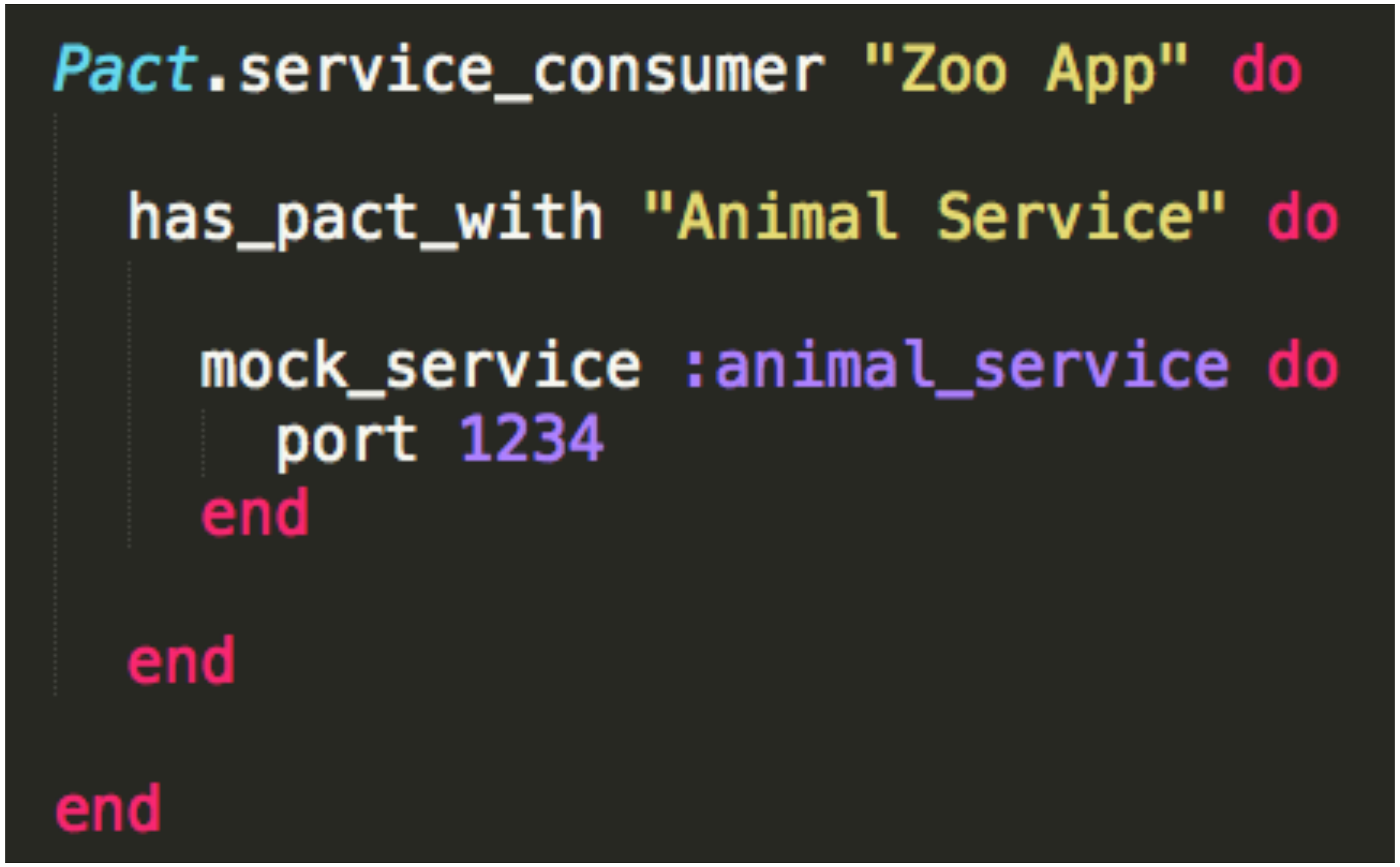
Pact… how it works



Problems with...	***Solved by Pact***
Test symmetry	***Test symmetry***
● ***No automated tools***	● ***Automatically enforced***
● ***Relies on developer eternal vigilance - not sustainable***	
● ***Does not scale***	
Integrated tests	***Integrated tests***
● ***Slow***	● ***Fast***
● ***Easy to break***	● ***Low set up***
● ***Hard to fix***	● ***Reliable***
● ***Scales combinatorially***	● ***Easier to debug***
	● ***Standalone CI build***
	● ***No extra infrastructure***
	● ***Scales linearly***

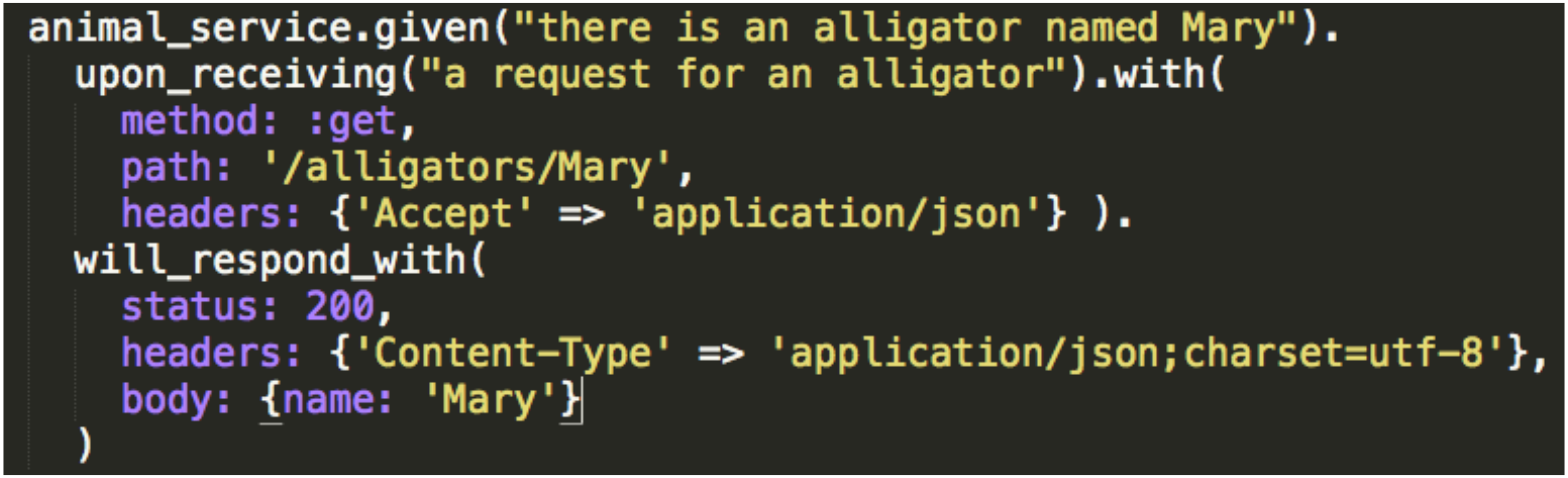
Better API design!

Arrange - set up mock server



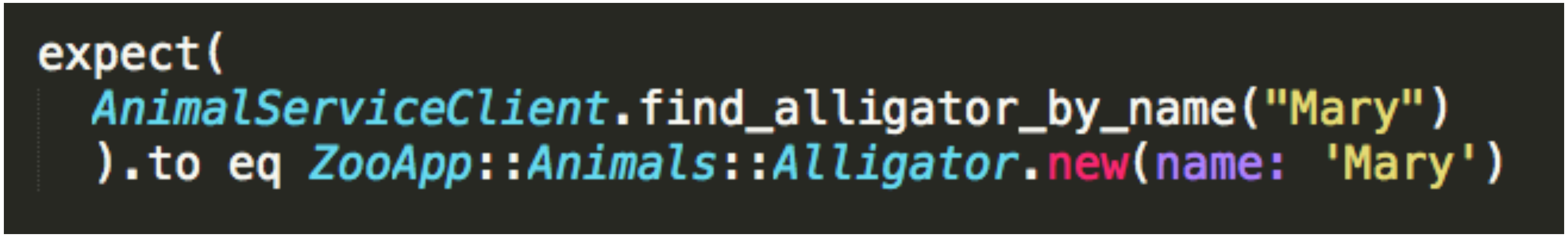
```ruby
Pact.service_consumer "Zoo App" do

  has_pact_with "Animal Service" do

    mock_service :animal_service do
      port 1234
    end

  end

end
```

Arrange - set up expectations

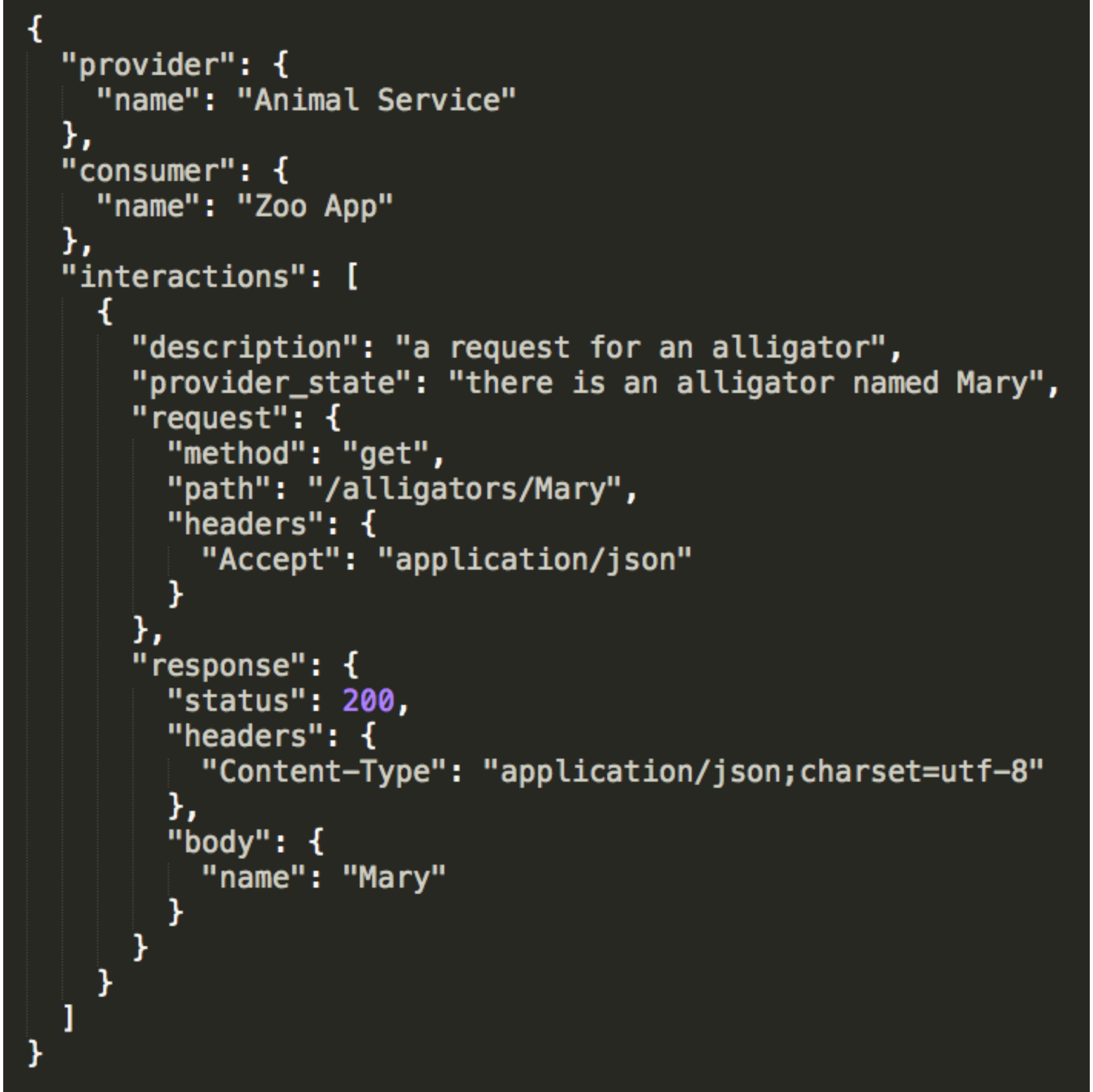


```
animal_service.given("there is an alligator named Mary").
  upon_receiving("a request for an alligator").with(
    method: :get,
    path: '/alligators/Mary',
    headers: {'Accept' => 'application/json'} ).
  will_respond_with(
    status: 200,
    headers: {'Content-Type' => 'application/json;charset=utf-8'},
    body: {name: 'Mary'}
  )
```

Act and Assert

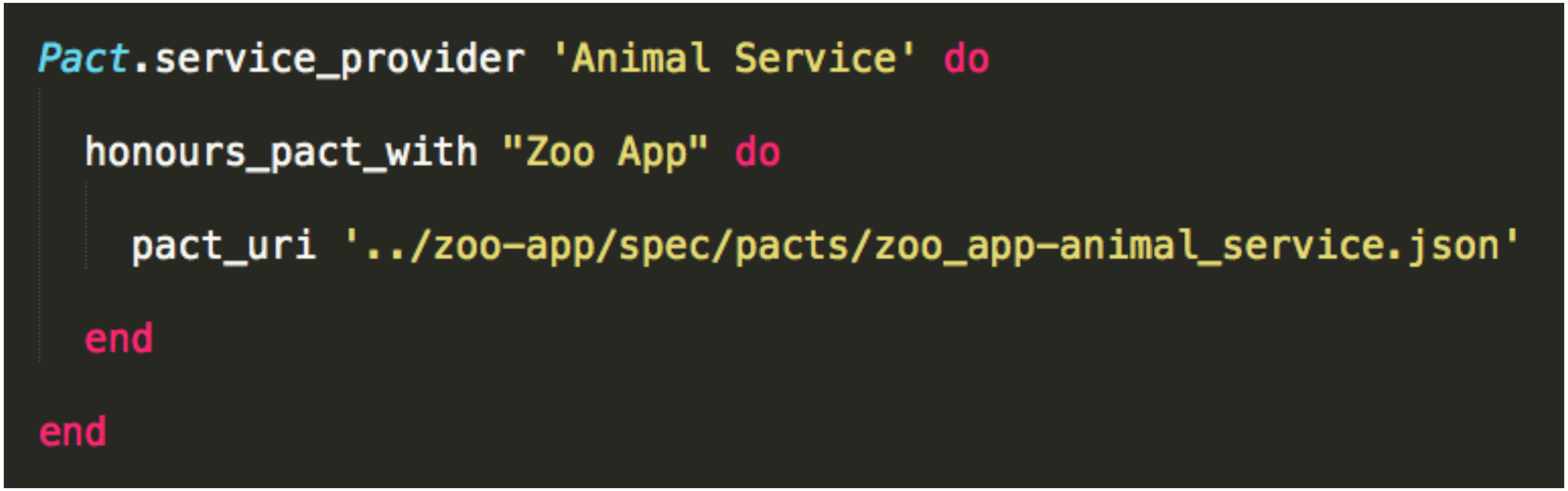


```
expect(
  AnimalServiceClient.find_alligator_by_name("Mary")
).to eq ZooApp::Animals::Alligator.new(name: 'Mary')
```



```json
{
  "provider": {
    "name": "Animal Service"
  },
  "consumer": {
    "name": "Zoo App"
  },
  "interactions": [
    {
      "description": "a request for an alligator",
      "provider_state": "there is an alligator named Mary",
      "request": {
        "method": "get",
        "path": "/alligators/Mary",
        "headers": {
          "Accept": "application/json"
        }
      },
      "response": {
        "status": 200,
        "headers": {
          "Content-Type": "application/json;charset=utf-8"
        },
        "body": {
          "name": "Mary"
        }
      }
    }
  ]
}
```

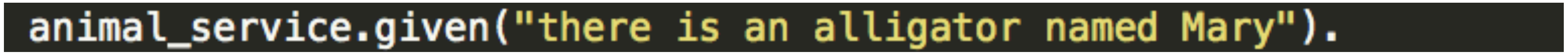
Set up provider



```ruby
Pact.service_provider 'Animal Service' do
  honours_pact_with "Zoo App" do
    pact_uri '../zoo-app/spec/pacts/zoo_app-animal_service.json'
  end
end
```

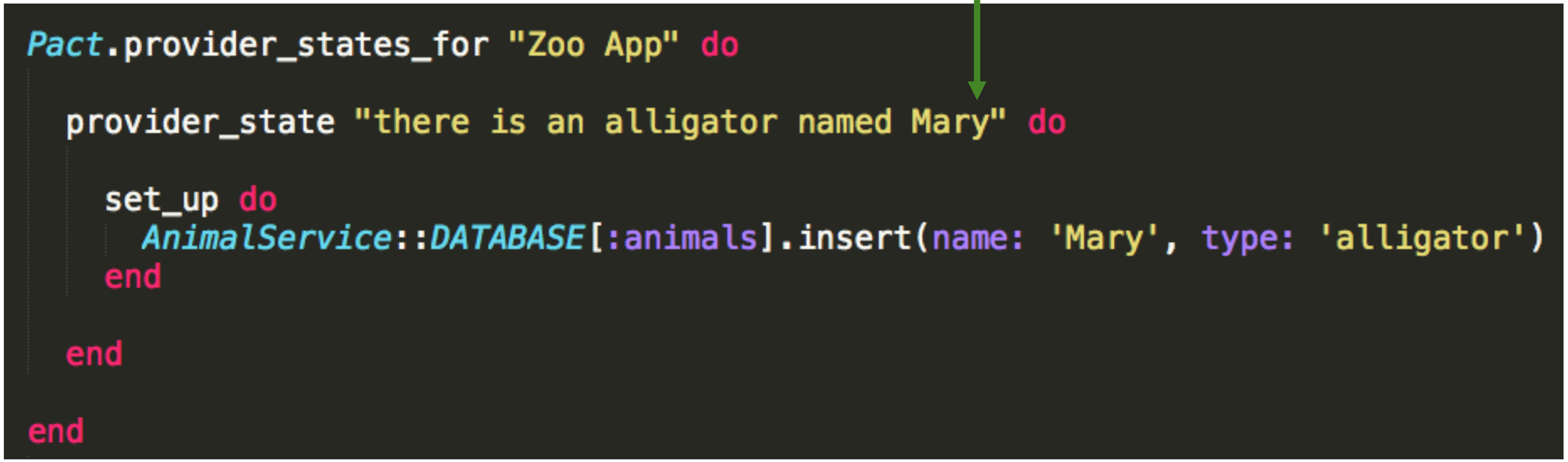
Set up test data

Consumer assumed:



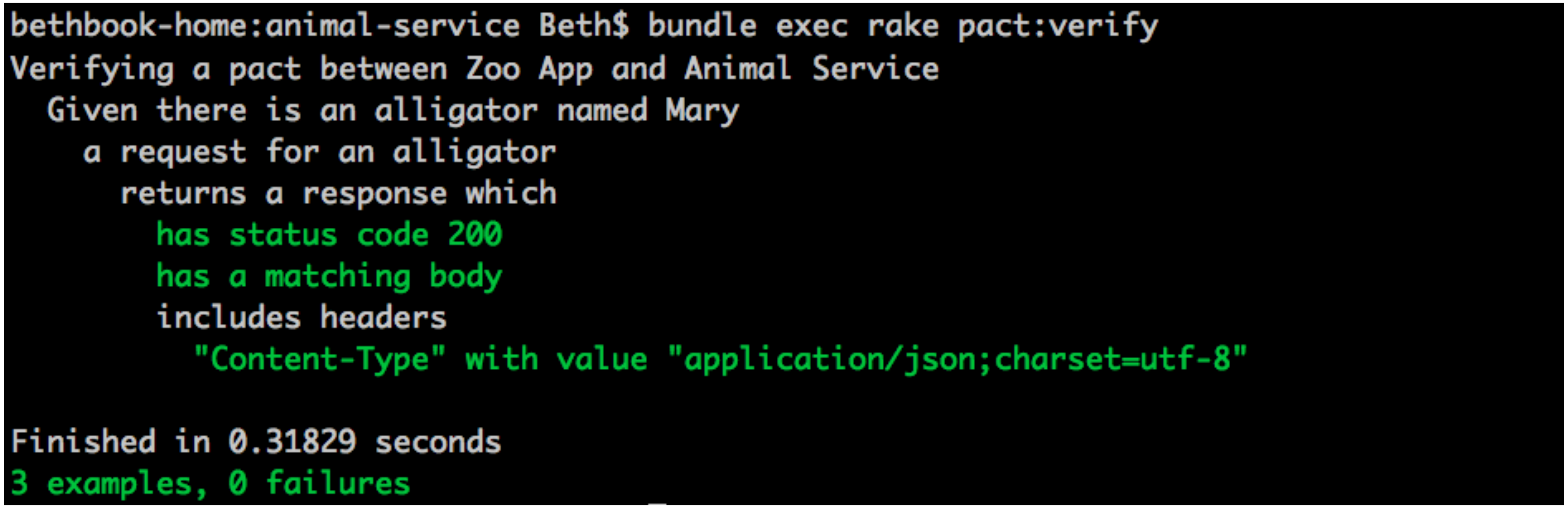
```
animal_service.given("there is an alligator named Mary").
```

Provider complies:



```
Pact.provider_states_for "Zoo App" do

  provider_state "there is an alligator named Mary" do

    set_up do
      AnimalService::DATABASE[:animals].insert(name: 'Mary', type: 'alligator')
    end

  end

end
```

Verify: Success!



```
bethbook-home:animal-service Beth$ bundle exec rake pact:verify
Verifying a pact between Zoo App and Animal Service
  Given there is an alligator named Mary
    a request for an alligator
      returns a response which
        has status code 200
        has a matching body
        includes headers
          "Content-Type" with value "application/json;charset=utf-8"

Finished in 0.31829 seconds
3 examples, 0 failures
```

Support

- JSON is language independent
- Ruby
- JVM (Java and Scala)
- Validate against any running provider using either library
- Pact specification
- JS library next - volunteers anyone?
- Janus
- Shokkenki

“How is it working for you?”

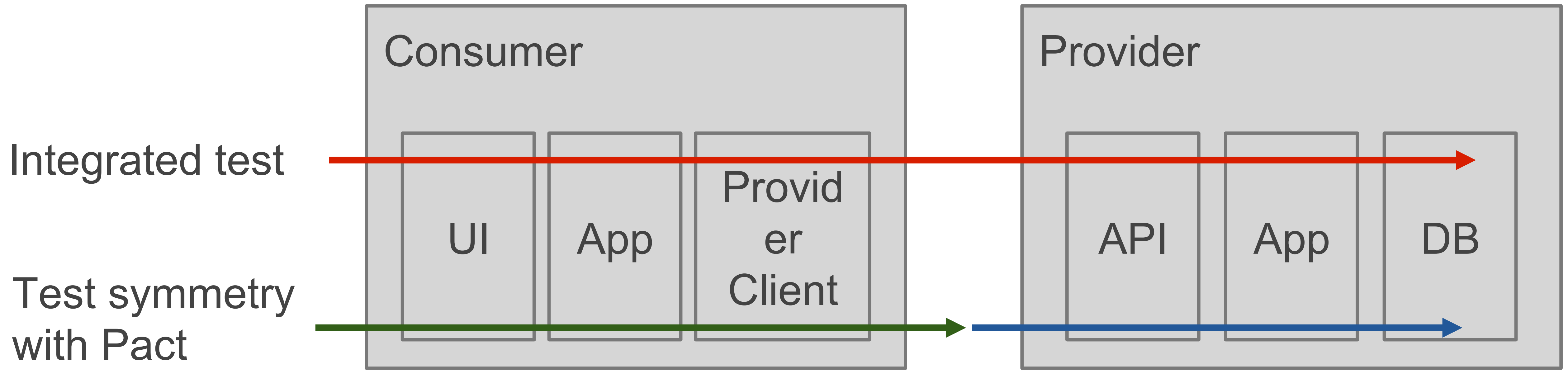
Before Pact

- A “certification” environment that took 2 months to set up
- Deploy all the things => deploy new thing => integration tests => pass ? deploy to production : try again
- Does not scale

With Pact

- 40+ microservices using Pact
- Green standalone CI builds ? deploy to production

In summary...


Consumer
Provider
Integrated test
UI
App
Provid
er
Client
API
App
DB
Test symmetry
with Pact

Pact allows you to:

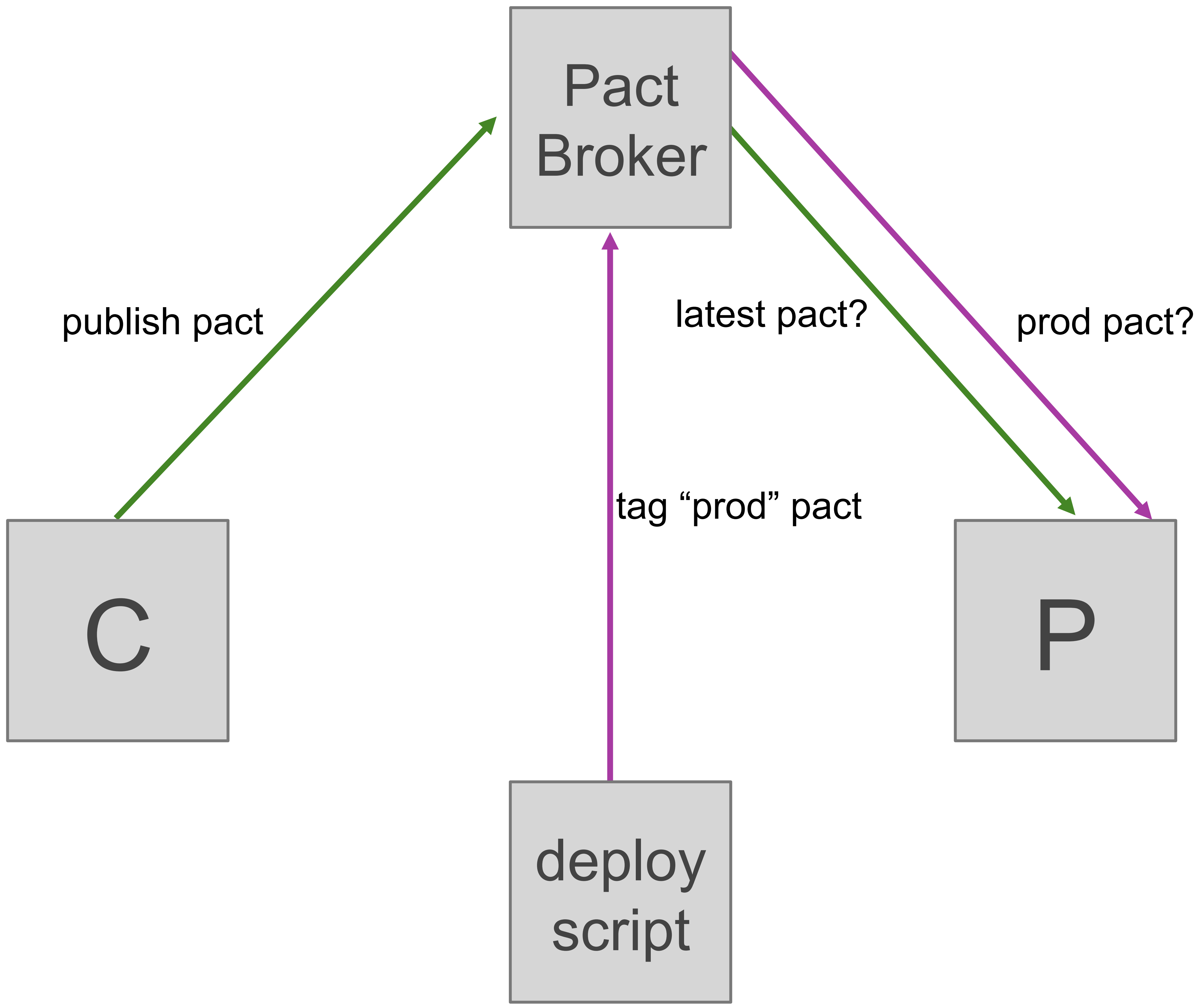
- Combine the advantages of integrated tests and test symmetry
- Mitigates the shortcomings of each method
- Allows you to modify components with agility - quick feedback on potential breakages
- Have confidence that all the services in your system will work together
- Throw away your integration tests!!!

Pact

Makes your microservices play nicely together
with Consumer Driven Contracts

github.com/realestate-com-au/pact
@pact_up

Beth Skurrie
bskurrie@dius.com.au
@bethesque


Pact Broker
publish pact
latest pact?
prod pact?
tag "prod" pact
C
P
deploy script