Testing via Jasmine

About Me
(why you should listen)

- @pseale on twitter
- Years of real testing experience
- I have nothing to lose

improving

It´s what we do.™

Roadmap

- Introduce Jasmine
- Show end-to-end testing with Selenium
- Community discussion

For discussion:

- Anything I say that is dumb
- Useful tips you have (interrupt!)
- “Testing is hard for me b/c (reason) ______”
- Experience report of commercial tool (e.g. SauceLabs)

For discussion:

- Testing your server-side via HTTP
- Unit testing significant chunks of your front-end (and does your **architecture** make it easy?)
- Selenium war stories
- Actual war stories

The best testing wisdom comes from experience

thus, community discussion

Roadmap

- **Introduce Jasmine**
- Show end-to-end testing with Selenium
- Community discussion

Demo 1: unit testing in Jasmine

Unit tests: demo recap

- fast
- help design/internal code quality
- small steps ease the brain burden
- refactor with confidence
- **app doesn’t work**

takeaway: use the correct assertions to give yourself good error messages

Roadmap

- Introduce Jasmine
- **Show end-to-end testing with Selenium**
- Community discussion

Demo 2: browser testing via Selenium

End-to-end tests: demo recap

- **app works** (“executable specifications”)
- refactor with confidence
- **slow**
- **setting up test environment is brutal**
- **setting up test data (the //Arrange section) is brutal & tests are brittle**
- **internal quality is garbo**

takeaway: using Selenium isn’t hard

Roadmap

- Introduce Jasmine
- Show end-to-end testing with Selenium
- **Community discussion**

Unit vs Integration

- fast
- help design/internal code quality
- small steps ease the brain burden
- refactor with confidence
- **app doesn’t work**

- **app works** (“executable specifications”)
- refactor with confidence
- **slow**
- **setting up test environment is brutal**
- **setting up test data (the // Arrange section) is brutal & tests are brittle**
- **internal quality is garbo**

How can we fix this?

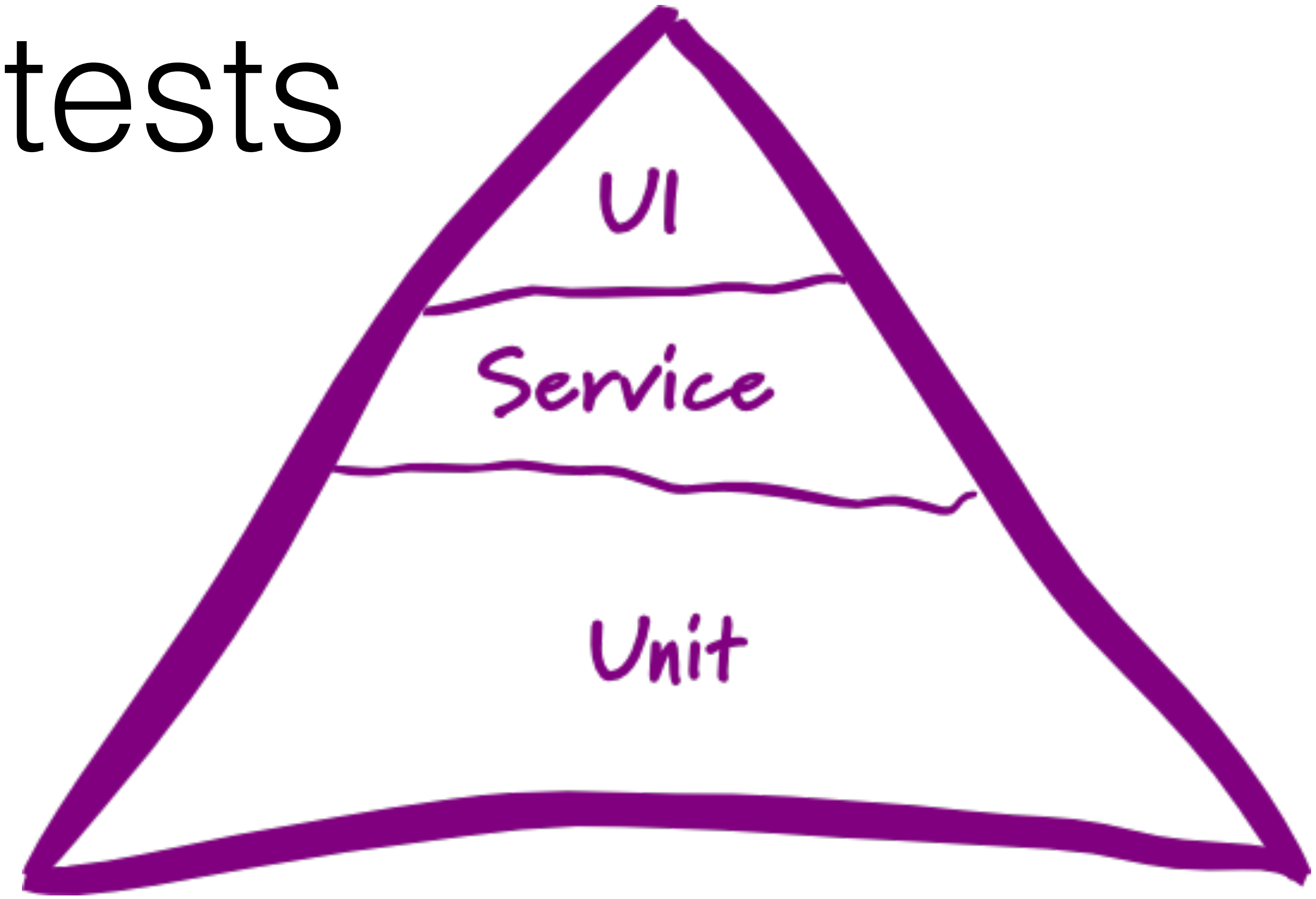
- **slow**

- **setting up test environment is brutal**

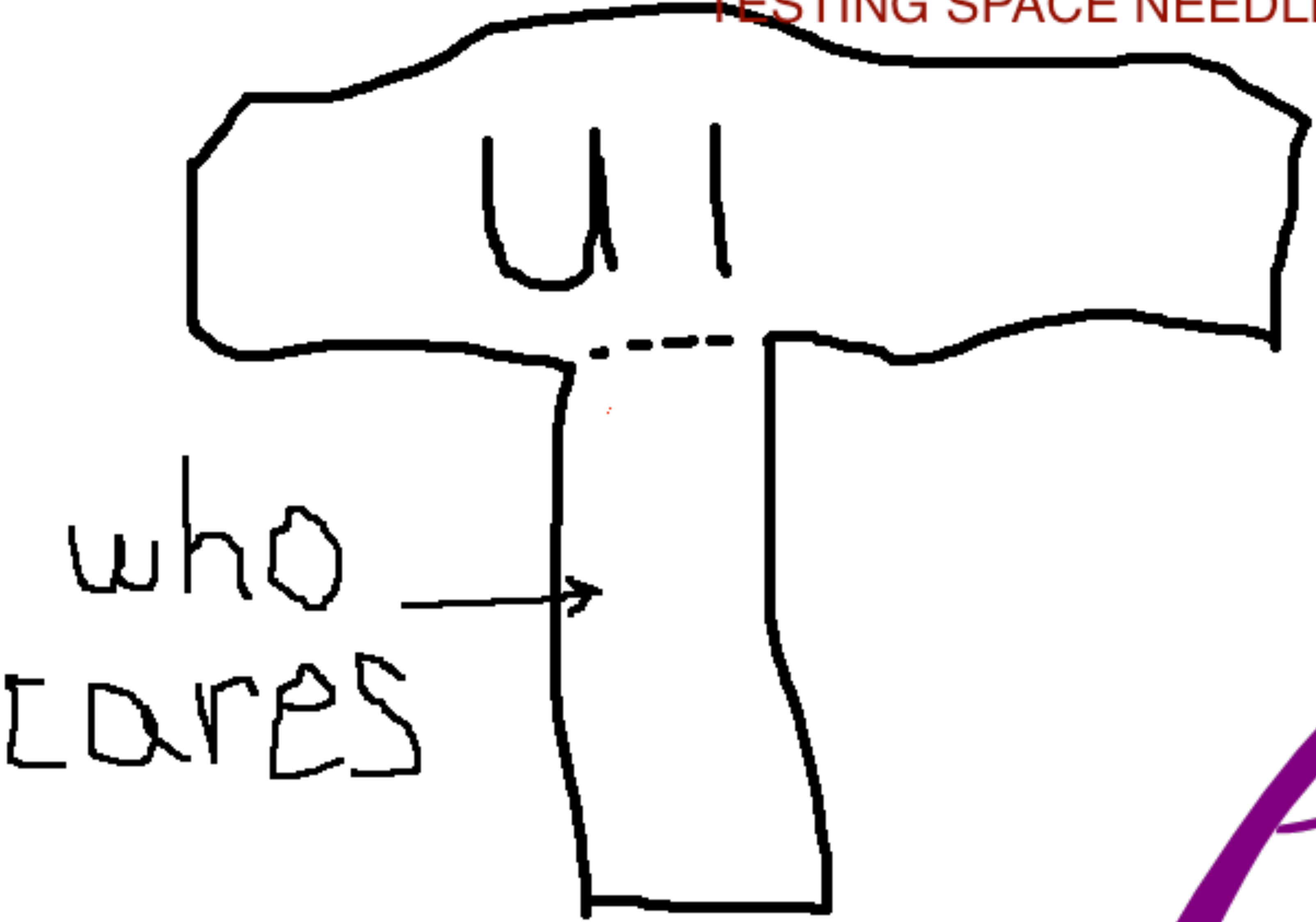
- **setting up test data (the //Arrange section) is brutal & tests are brittle**

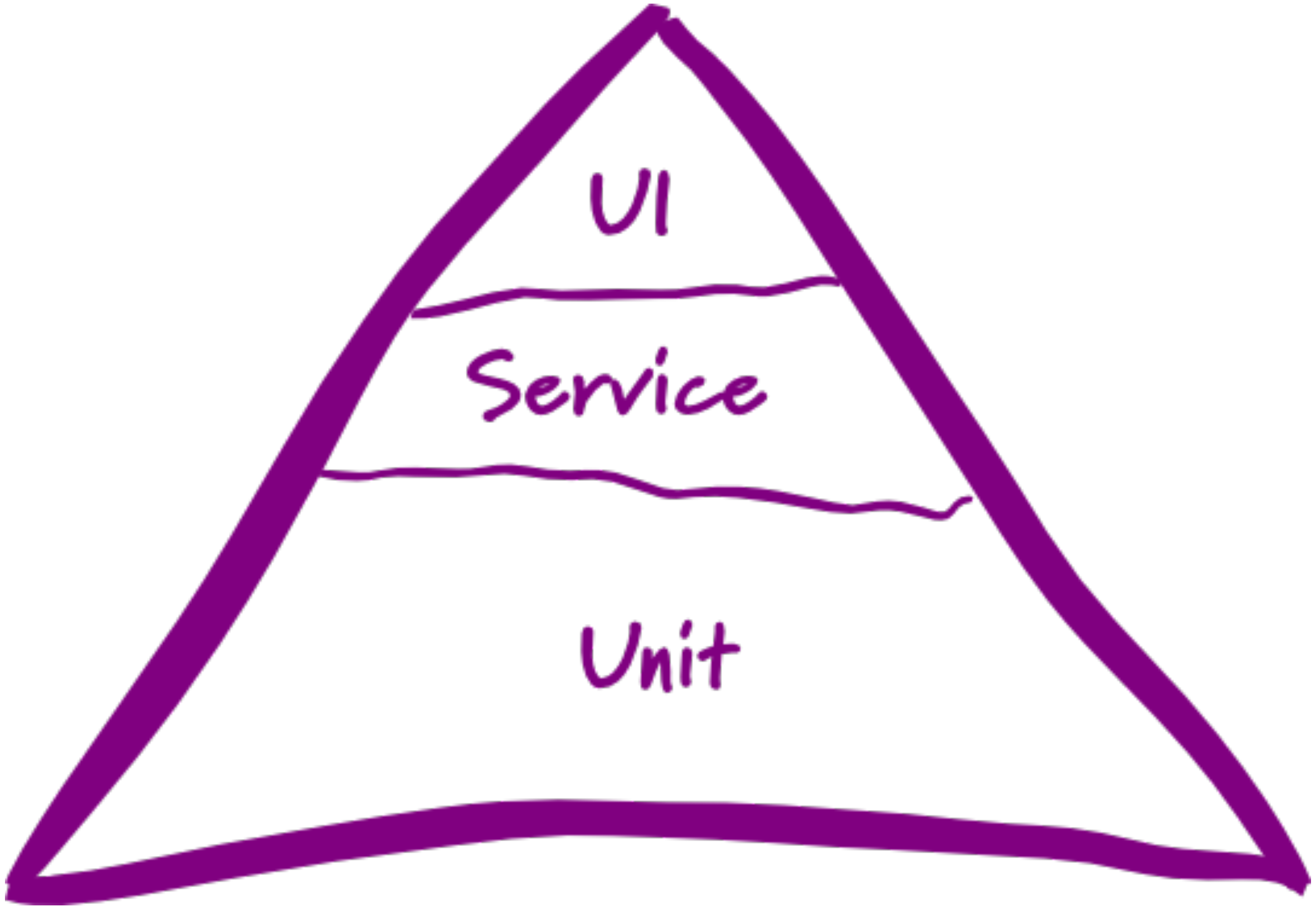
- **internal quality is garbo**

Fix **internal quality**: write both unit & integration tests



TESTING SPACE NEEDLE





Fix the problem of setting up test data for integration tests:
test builders

Test builder:
bad examples

```
FactoryGirl.create :user,
  :twitter => ‘@pseale’,
  :email => “http:/website.biz”
```

```
var order = {
  lineItems: [
    {name:’peanut butter’, cost:5.00},
   {name:’jelly’, cost:5.00}],
  grandTotal: 11.00
}
```

“If your test helpers can create garbage data, they’re useless. And you’re useless as well.”

-@pseale’s Tenth Law

Test helpers: slide-as-demo

```
FactoryGirl.create :user,
  :twitter => ‘@pseale’,
  :email => “http:/website.biz”
```

```
CreateUserCommand.execute(
  ‘@pseale’,
  ‘http:/website.biz’)
==> error: invalid email
```

```
var order = {
  lineItems: [
    {name:’peanut butter’,
     cost:5.00},
   {name:’jelly’, cost:5.00}],
  grandTotal: 11.00
}
```

```
var order = new OrderBuilder().
  addLineItem(‘peanut butter’, 5.00).
  addLineItem(‘jelly’, 5.00).
  build()
```

Solving common Selenium problems

use Page Objects | wait for AJAX requests to finish | disable CSS animations | stub out external APIs | cheat | give up

Discussion point: how can we write better **unit** tests?

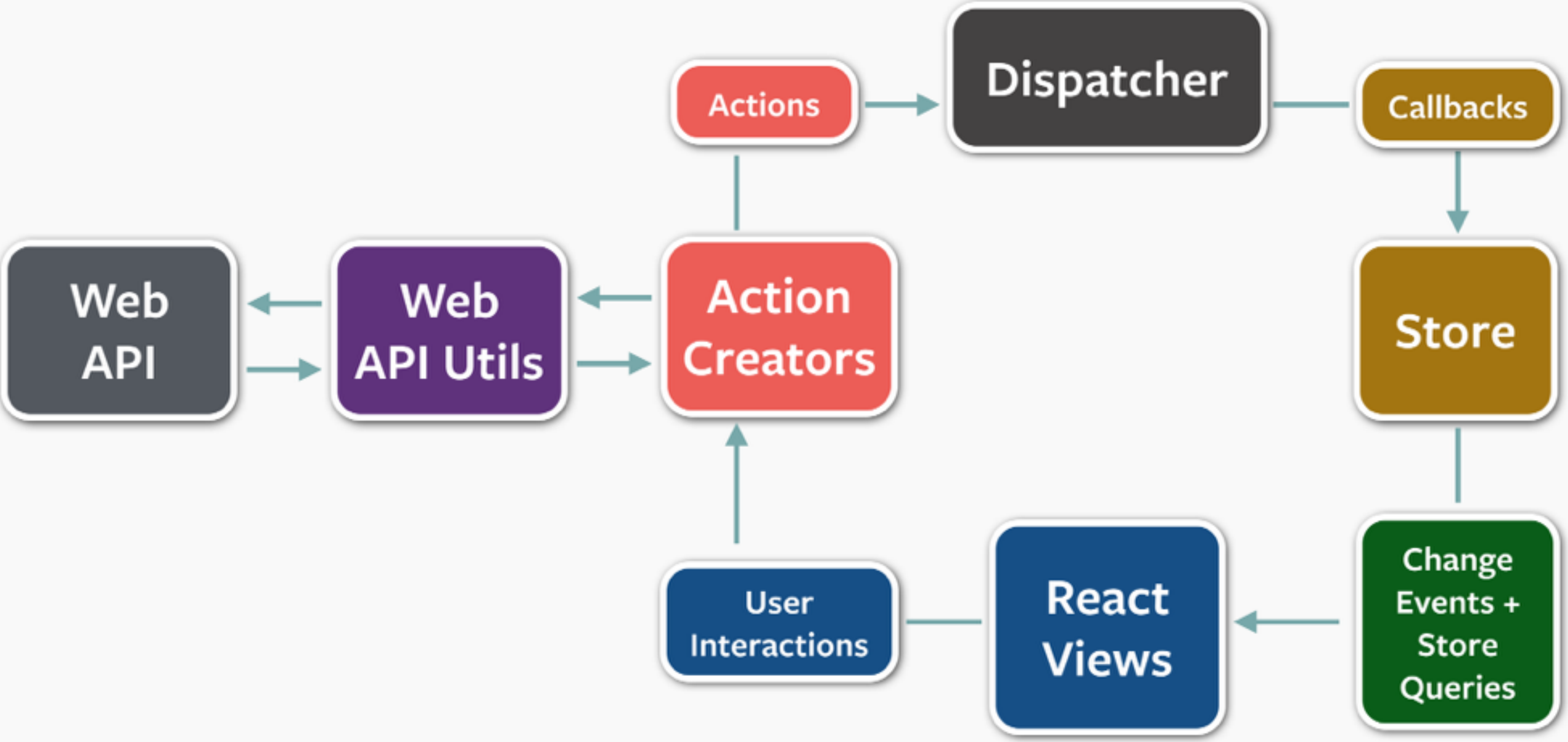
I have never figured it out **in practice**

Discussion:
test helpers (tips?)

For discussion:

- Testing your server-side via HTTP
- Unit testing significant chunks of your front-end
- Selenium war stories
- Actual war stories

Discussion topic:
mitigating slow tests


Actions
Dispatcher
Callbacks
Web API
Web API Utils
Action Creators
Store
User Interactions
React Views
Change Events + Store Queries

Discussion topic:

the END of SHAME

(AKA Testing Nudism)

Discussion topic:
stop making new assertion libraries,
they’re getting worse