

Snapshot testing on Sister

What is it?

A typical snapshot test case, takes a snapshot, then compares it to a reference snapshot file stored alongside the test.

An example (Swift)

```
func testLoadedScreen() {  
    // Given  
    Current.apiClient.mock(  
        request: HelpRequest.GetForm(identifier: "foo"),  
        response: .success(.json("HelpFormWithoutInitialValueResponse"))  
    )  
    let sut = ReportProblemViewController(identifier: "foo")  
  
    // Then  
    assertSnapshot(matching: sut)  
    assertSnapshot(matching: sut.store.state)  
}
```

UI snapshot

```
func testLoadedScreen() {  
    // Given  
    Current.apiClient.mock(  
        request: HelpRequest.GetForm(identifier: "foo"),  
        response: .success(.json("HelpFormWithoutInitialValueResponse"))  
    )  
    let sut = ReportProblemViewController(identifier: "foo")  
  
    // Then  
    assertSnapshot(matching: sut)  
    assertSnapshot(matching: sut.store.state)  
}
```

Describe with more details the problem you are experiencing.

Describe your problem

Your problem

0 / 10 min

Document example

I, **name and surname**, present my voluntary resignation.

Your last day

Select a date

Send a picture of the signed document

The picture shouldn't be blurred



Add picture

Reminder

Send

State snapshot

```
func testLoadedScreen() {
    // Given
    Current.apiClient.mock(
        request: HelpRequest.GetForm(identifier: "foo"),
        response: .success(.json("HelpFormWithoutInitialValueResponse"))
    )
    let sut = ReportProblemViewController(identifier: "foo")

    // Then
    assertSnapshot(matching: sut)
    assertSnapshot(matching: sut.store.state)
}
```

```
▼ ReportProblemViewState
  ▼ loadingState: LoadingState<FormViewState, FormViewStateError>
  ▼ loaded: FormViewState
    - title: "Problem details"
    ▼ summary: Optional<String>
      - some: "Describe with more details the problem you are experiencing."
    ▼ fieldsState: 6 elements
      ▼ FieldState
        - header: "Describe your problem"
      ▼ FieldState
        ▼ textInput: TextInputState
          ▼ identifier: description
            - rawValue: "description"
            - placeholder: "Your problem"
          ▼ validation: AnimatedTextInputValidation
            ▼ both: ClosedRange(10...320)
              - lowerBound: 10
              - upperBound: 320
            - text: Optional<String>.none
            - error: Optional<String>.none
        ▼ FieldState
          - header: "Document example"
        ▼ FieldState
          - sampleText: "I, <b>name and surname</b>, present my voluntary resignation."
        ▼ FieldState
          - header: "Your last day"
        ▼ FieldState
          ▼ dateInput: DateInputState
            ▼ identifier: last-day-date
              - rawValue: "last-day-date"
              - placeholder: "Select a date"
              - date: Optional<Date>.none
          - isSendingForm: false
```

A typical snapshot test case:

1. Creates the reference snapshot file stored alongside the test
2. Takes a snapshot
3. Compares it to the reference snapshot
4. The test will fail if the two snapshots do not match:
 - either the change is unexpected
 - or the reference snapshot needs to be updated to the new version of the UI component or state.

JobandtalentKit

iPhone 8

jobandtalent | Build JobandtalentKit: Succeeded | Today at 09:09

3 1

jobandtalent > JobandtalentKit > Sources > Report > Form > ReportProblemViewControllerTests.swift > testLoadedScreen()

< > < > < >

✖

```
func testLoadedScreen() {  
    // Given  
    Current.apiClient.mock(  
        request: HelpRequest.GetForm(identifier: "foo"),  
        response: .success(.json("HelpFormWithoutInitialValueResponse"))  
    )  
    let viewController = ReportProblemViewController(identifier: "foo")  
  
    // Then  
    assertSnapshot(matching: viewController)  
    assertSnapshot(matching: viewController.store.state)  
    assertSnapshot(matching: Current.navigator.h  
    assertSnapshot(matching: Current.analyticsTr  
}  
  
func testGenericErrorScreen() {  
    // Given  
    Current.apiClient.mock(  
        request: HelpRequest.GetForm(identifier:  
        response: .failure(.error(.sessionExpire  
    )  
    let viewController = ReportProblemViewContr
```

◇

failed - Diff: ...

@@ -48,13 +48,6 @@
- _url: <https://jobandtalent.zepeling.com/resource/bill-icon.jpg>
- isSendingForm: false
- identifier: foo
- rawValue: "foo"
- context: Optional<HelpContext>
- some: HelpContext
- context: 1 key/value pair
- (2 elements)
- key: "breadcrumb"
- value: 2 elements
- "topic"
- "foo"



failed - Diff: ...



@@ -48,13 +48,6 @@

- _url: <https://jobandtalent.zepeling.com/resource/bill-icon.jpg>

- isSendingForm: false

▽ identifier: foo

- rawValue: "foo"

- ▽ context: Optional<HelpContext>

- ▽ some: HelpContext

- ▽ context: 1 key/value pair

- ▽ (2 elements)

- - key: "breadcrumb"

- ▽ value: 2 elements

- - "topic"

- - "foo"

+ - context: Optional<HelpContext>.none

```
code --diff "/Users/jobandtalent/Development/ios-  
jobandtalent/JobandtalentKit/Sources/Report/Form/  
__Snapshots__/ReportProblemViewControllerTests/  
testLoadedScreen.1.txt" "/Users/jobandtalent/Library/  
Developer/CoreSimulator/Devices/  
87E0C864-03EB-48A1-9135-022B1CAA437B/data/tmp/  
testLoadedScreen.1.txt"
```


Snapshot testing on Sister (Elixir)

Before snapshot testing

```
test "lists shifts from a worker candidate_id", %{conn: conn} do
  conn =
    get(
      conn,
      Routes.shift_path(conn, :index,
        job_id: "wrk_candidate_id:some-id",
        auth_token: "valid-worker"
      )
    )

  response = json_response(conn, 200)
  assert_schema("worker/shifts/list_response.json", response)

  [shift] = response["data"]
  assert "12345" == shift["id"]
  assert "22:30 - 02:30" == shift["description"]
  assert "2018-09-21T22:30:00" == shift["local_start_at"]
  assert "2018-09-22T02:30:00" == shift["local_end_at"]
  assert "Europe/Madrid" == shift["local_tz"]
  assert [] == shift["breaks"]
  refute Map.has_key?(shift, "breaks_scheme")
  assert "pending" == shift["status"]

  assert [] = shift["accepted_responses"]

  location = shift["location"]
  assert "Barcelona" == location["full_address"]
  assert "Barcelona" == location["description"]
  assert 41.5 == location["lat"]
  assert -1.0 == location["lng"]
end
```

Asserting the controller output

```
test "lists shifts from a worker candidate_id", %{conn: conn} do
  conn =
    get(
      conn,
      Routes.shift_path(conn, :index,
        job_id: "wrk_candidate_id:some-id",
        auth_token: "valid-worker"
      )
    )

  response = json_response(conn, 200)
  assert_schema("worker/shifts/list_response.json", response)

  [shift] = response["data"]
  assert "12345" == shift["id"]
  assert "22:30 - 02:30" == shift["description"]
  assert "2018-09-21T22:30:00" == shift["local_start_at"]
  assert "2018-09-22T02:30:00" == shift["local_end_at"]
  assert "Europe/Madrid" == shift["local_tz"]
  assert [] == shift["breaks"]
  refute Map.has_key?(shift, "breaks_scheme")
  assert "pending" == shift["status"]

  assert [] = shift["accepted_responses"]

  location = shift["location"]
  assert "Barcelona" == location["full_address"]
  assert "Barcelona" == location["description"]
  assert 41.5 == location["lat"]
  assert -1.0 == location["lng"]
end
```

With snapshot testing

```
test "lists shifts from a worker candidate_id", %{conn: conn} do
  conn =
    get(
      conn,
      Routes.shift_path(conn, :index,
        job_id: "wrk_candidate_id:some-id",
        auth_token: "valid-worker"
      )
    )

  response = json_response(conn, 200)
  assert_schema("worker/shifts/list_response.json", response)
  match_snapshot response["data"]
end
```

The assertion

```
test "lists shifts from a worker candidate_id", %{conn: conn} do
  conn =
    get(
      conn,
      Routes.shift_path(conn, :index,
        job_id: "wrk_candidate_id:some-id",
        auth_token: "valid-worker"
      )
    )

  response = json_response(conn, 200)
  assert_schema("worker/shifts/list_response.json", response)
  match_snapshot response["data"]
end
```

Running the tests

```
jobandtalent@caldofran: ~/Development/sister
~/Development/sister } elixir-snapshot-testing ➔ clear
~/Development/sister } elixir-snapshot-testing ● ➔ mix test

.....*
.....*
.....S.....*
.....

Finished in 4.2 seconds
8 doctests, 912 tests, 0 failures, 2 skipped

Randomized with seed 125086
~/Development/sister } elixir-snapshot-testing ● ➔ mix test

.....*
.....*
.....

Finished in 4.3 seconds
8 doctests, 912 tests, 0 failures, 2 skipped

Randomized with seed 556704
~/Development/sister } elixir-snapshot-testing ● |
```

The output file

test_index_lists_shifts_from_a_worker_candidate_id.snap

```
[
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "12345",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  }
]
```

Failing test

```
jobandtalent@caldofran: ~/Development/sister
~/Development/sister ➤ elixir-snapshot-testing ➤ mix test test/sister_web/controllers/shifts/shift_controller_test
.exs:409
Excluding tags: [:test]
Including tags: [line: "409"]

1) test index lists shifts from a worker candidate_id (SisterWeb.Shifts.ShiftControllerTest)

    Received value does not match stored snapshot. (__snapshots__/sister_web/controllers/shifts/shift_controller_test/
    test_index_lists_shifts_from_a_worker_candidate_id.snap)
      code: Snapshot == Received
      left: [%{"accepted_responses" => [], "breaks" => [], "local_start_at" => "2018-09-21T22:30:00", "local_tz" => "Eu
      rope/Madrid", "status" => "pending", "description" => "22:30 - 02:30", "id" => "12345", "local_end_at" => "2018-09-22T0
      2:30:00", "location" => %{"lat" => 41.5, "lng" => -1, "description" => "Barcelona", "full_address" => "Barcelona"}}]
      right: [%{"accepted_responses" => [], "breaks" => [], "local_start_at" => "2018-09-21T22:30:00", "local_tz" => "Eu
      rope/Madrid", "status" => "pending", "description" => "22:30 - 09:20", "id" => "123456789", "local_end_at" => "2018-09-
      22T09:20:00", "location" => %{"lat" => 41.5, "lng" => -1, "description" => "Madrid", "full_address" => "Madrid"}}]
      stacktrace:
        (snapshy) lib/snapshy.ex:136: Snapshy.raise_error/3
        (snapshy) lib/snapshy.ex:107: Snapshy.match/2
        test/sister_web/controllers/shifts/shift_controller_test.exs:409: (test)

Finished in 0.3 seconds
19 tests, 1 failure, 18 excluded

Randomized with seed 536685
~/Development/sister ➤ elixir-snapshot-testing ➤
```


Traditional

```
[shift] = response["data"]
assert "12345" == shift["id"]
assert "22:30 - 02:30" == shift["description"]
assert "2018-09-21T22:30:00" == shift["local_start_at"]
assert "2018-09-22T02:30:00" == shift["local_end_at"]
assert "Europe/Madrid" == shift["local_tz"]
assert [] == shift["breaks"]
refute Map.has_key?(shift, "breaks_scheme")
assert "pending" == shift["status"]

assert [] = shift["accepted_responses"]

location = shift["location"]
assert "Barcelona" == location["full_address"]
assert "Barcelona" == location["description"]
assert 41.5 == location["lat"]
assert -1.0 == location["lng"]
```

Snapshot

```
[
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "12345",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  }
]
```

Advantages

- Easy to create
- Simpler and less noisy assertion
- **Broader coverage with very little effort**

Disadvantages

- The assertion is not in the test
- Any change in the output, the snapshot must be regenerated
- Not TDD
- Could slow down the suite test
- Few libraries in Elixir ecosystem:
 - Snapshoty, response snapshot

Ruby

- Approvals
- rspec-snapshot

Why in Sister

Simpler integration tests

**modules + controllers + views +
translations + assets**

Actually testing view state

Describe with more details the problem you are experiencing.

Describe your problem

Your problem

0 / 10 min

Document example


I, **name and surname**, present my voluntary resignation.

Your last day

Select a date

Send a picture of the signed document

The picture shouldn't be blurred



Add picture

Reminder

Send

```
{
  "data": {
    "id": "bar",
    "type": "form",
    "title": "Problem details",
    "summary": "Describe with more details the problem you are experiencing.",
    "items": [
      {
        "type": "section-title",
        "label": "Describe your problem"
      },
      {
        "type": "text-area",
        "label": "Your problem",
        "param_name": "description"
      },
      {
        "type": "text-area",
        "label": "Something else",
        "param_name": "additional",
        "validations": {
          "min_length": 20,
          "max_length": 255
        },
        "initial_value": "I have nothing else to add"
      }
    ],
    "context": [
      ["breadcrumb[]", "topic"],
      ["breadcrumb[]", "foo"]
    ]
  }
}
```

More secure

Broader coverage

Schema tests are not enough

```
test "lists shifts from a worker candidate_id", %{conn: conn} do
  conn =
    get(
      conn,
      Routes.shift_path(conn, :index,
        job_id: "wrk_candidate_id:some-id",
        auth_token: "valid-worker"
      )
    )

  response = json_response(conn, 200)
  assert_schema("worker/shifts/list_response.json", response)
  match_snapshot response["data"]
end
```

This is valid schema

```
[
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "12345",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  }
]
```

This one is valid too

```
[
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "12345",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  },
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "6789",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  }
]
```

The whole controller output is checked

```
[
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "12345",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  },
  %{
    "accepted_responses" => [],
    "breaks" => [],
    "description" => "22:30 - 02:30",
    "id" => "6789",
    "local_end_at" => "2018-09-22T02:30:00",
    "local_start_at" => "2018-09-21T22:30:00",
    "local_tz" => "Europe/Madrid",
    "location" => %{
      "description" => "Barcelona",
      "full_address" => "Barcelona",
      "lat" => 41.5,
      "lng" => -1
    },
    "status" => "pending"
  }
]
```

Traditional would not catch the error

```
[shift] = response["data"]
assert "12345" == shift["id"]
assert "22:30 - 02:30" == shift["description"]
assert "2018-09-21T22:30:00" == shift["local_start_at"]
assert "2018-09-22T02:30:00" == shift["local_end_at"]
assert "Europe/Madrid" == shift["local_tz"]
assert [] == shift["breaks"]
refute Map.has_key?(shift, "breaks_scheme")
assert "pending" == shift["status"]

assert [] = shift["accepted_responses"]

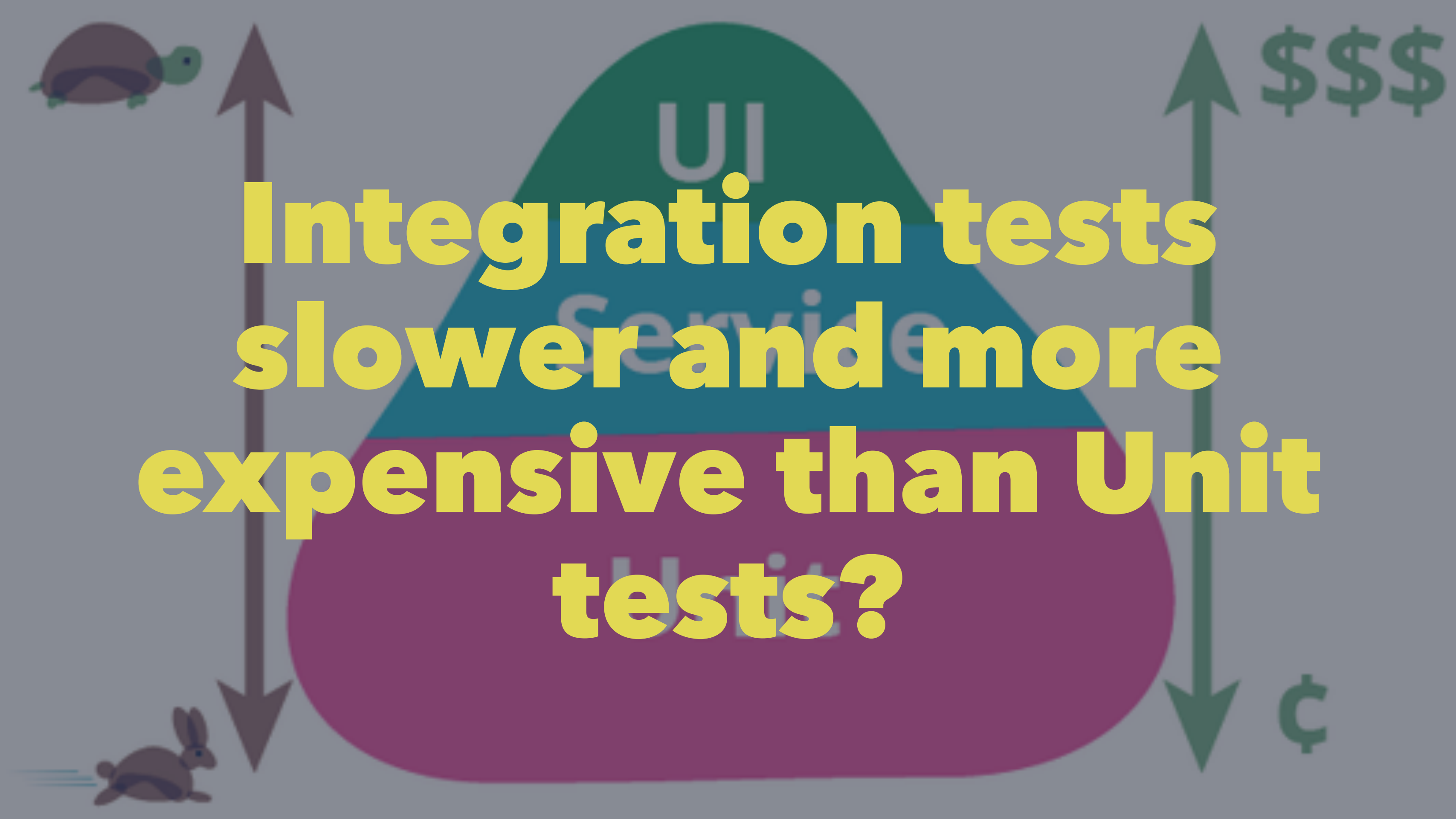
location = shift["location"]
assert "Barcelona" == location["full_address"]
assert "Barcelona" == location["description"]
assert 41.5 == location["lat"]
assert -1.0 == location["lng"]
```

Snapshot testing

DOES NOT

replace Unit testing

**Core logic should be tested via
unit tests in the modules**



**Integration tests
slower and more
expensive than Unit
tests?**

Follow up

<https://github.com/jobandtalent/sister/pull/409>

? ? ? **or troll me**

*A good brother takes care
of Sister*

– Caldofran