Elixir testing - Tag tests in ExUnit

21 APRIL 2015 on Elixir, ExUnit, Test

Scope tests in ExUnit to reduce code duplication and have cleaner code.

Update July 2016! This blog post is old, and things change. ExUnit now supports grouping with `describe`. Read more here.

Rspec nesting

People with ruby, and rspec, background use `describe` to group tests that are similar or require the same setup. You can nest `describe` so all tests in the inner describe have the same setup.

```
describe "User" do
  describe ".top" do
    before { FactoryGirl.create_list(:user, 3) }
    it { expect(User.top(2)).to have(2).item }
  end
end
```

Example taken from betterspecs.org.

This allows us to specify conditions (`before`) that are executed for all `it` inside `.top` but not any other specs.

Pretty handy.

ExUnit @tag

Elixirs ExUnit does not provide grouping tests in a module using a `dsl` like rspec. Instead we should use `@tag`.

`@tag` is quite powerful.

Timeout

A first a simple example using tags.

Given we have a test we expect to fail due to a timeout running the test will take the default 3000ms to fail. Using tags we can change this using `@tag timeout: 30` and the test fails way faster.

Excluding tests

Assume we have lots of tests, some depending on an external api (not mocked) and some that do not depend on the service. Running all tests would take time. **External APIs are slow by design.** Or imagine the service is down/you do not have access to the a connection then the tests will always fail.

Failing tests are a pain when developing new features. **Which failure is your fault?** So to clean this up you only want to run tests that do not require an external source. This can be achived using `@tag`.

For each test requiring a network connection add the tag `external: true`, and when running the tests add `--exclude external`.

```
mix test --exclude external
Excluding tags: [:external]

...............

Finished in 1.0 seconds (0.1s on load, 0.8s on tests)
15 tests, 0 failures
```

You could configure ExUnit to always exclude the external tag. Modify `test_helper.exs`:

```
ExUnit.configure exclude: [external: true]
ExUnit.start
```

When you want to run all tests use `--include external`.

```
mix test --include external:true
Including tags: [external: "true"]
Excluding tags: [external: true]

................

Finished in 1.0 seconds (0.1s on load, 0.8s on tests)
```

Grouping tests

We start with an example:

```
setup context do
  before(context)
  :ok
end

defp before(%{token: token}) do
  %Token{name: "token1", relic: token}
  |> Repo.insert
end

defp before(_params) do
end

test "assert no saved tokens" do
  assert 0 == Repo.all(Token) |> Enum.count
end

@tag token: "A_TOKEN"
test "assert one saved tokens" do
  assert 1 == Repo.all(Token) |> Enum.count
end

@tag token: "MyToken"
test "can access value from :token", context do
  assert "MyToken" == context[:token]
end
```

Running these using mix

```
mix test
...
```

```
Finished in 0.1 seconds (0.07s on load, 0.1s on tests)
3 tests, 0 failures
```

Here we have 3 tests. The first one have no tag, and thus no Token. It has neither a token value, nor have it been saved to the persistence layer. The other tests use `@tag token: <string>` and thus the before clause matches, a token is stored and in the test we can access the token value.

To access the tag value we use `context`which is passed to the test as such `test "some string", context do`.

Ofcource we can use the token tag så exclude all tests that dependece on it.

```
mix test --exclude token
Excluding tags: [:token]
.
Finished in 0.1 seconds (0.07s on load, 0.09s on tests)
1 tests, 0 failures
```

Conclusion

ExUnit `tag` is very nice. But I'd like to be able to group tests in a similar way as you can with rspec. I think its way easier to follow. Maybe that is just a simple little macro, but that is way over my head.

Happy coding.

Simon Ström

Share this post

Read more posts by this author.

READ THIS NEXT

Postfix - Redirect all outgoing mails to local user

While developing an application I often have to test, and use the email feature. Either I register and use...

YOU MIGHT ENJOY

Problems deploying Phoenix to Heroku

I tried to deploy a sample Phoenix application to Heroku so I read some blogposts about it etc. Didn't...