XMPP testing with Escalus

Krzysztof Goj

March 2, 2012

What is Escalus?

Escalus is a library for acceptance testing XMPP servers.

some code that makes doing certain things **easier**

What is Escalus?

Escalus is a library for acceptance testing XMPP servers.

checking if a thing **does what it's meant to**

What is Escalus?

Escalus is a library for acceptance testing XMPP servers.

XMPP (Jabber) - eXtensible Messaging and Presence Protocol

Background

- XMPP is an extensible protocol
- Our job is often to extend or adjust ejabberd to fit customer's demand.
- Ejabberd is a generic software - optimize by throwing away stuff you don't need.
- How to make sure we did the right thing?
- How to know if we break something as we go?

Before Escalus

Boring stuff is boring:

- Use Client's (say, Psi's) XMPP console.
- Works good for simple scenarios
- PITA with more complicated scenarios:
    - several users,
    - several resources per user,
    - setting up & tearing down the state.

after Escalus

Automated testing makes life easier:

- Continuous Integration
- TDD
- Set some tracing and replay the test.
- Get Wireshark dump – write a testcase.

Escalus test case

```
% Alice sends a chat message to Bob's bare JID
% Bob gets the message on both resources
% Bob replies to one of Alice's resources
% Alice receives the reply only on that resource
```

Escalus test case

```
message_routing(Config) ->
    escalus:story(Config, [2, 2],
        fun(Alice1, Alice2, Bob1, Bob2) ->
            % Alice sends a chat message to Bob's bare JID
            escalus:send(Alice1, escalus_stanza:chat_to(bob, <<"Coffee?">>),

            % Bob gets the message on both resources
            escalus:assert(is_chat_message, [<<"Coffee?">>],
                           escalus:wait_for_stanza(Bob1)),
            escalus:assert(is_chat_message, [<<"Coffee?">>],
                           escalus:wait_for_stanza(Bob2)),

            % Bob replies to one of Alice's resources
            escalus:send(Bob1, escalus_stanza:chat_to(Alice1, <<"Sure!">>),

            % Alice receives the reply only on that resource
            escalus:assert(is_chat_message, [<<"Sure!">>],
                           escalus:wait_for_stanza(Alice1)),
            escalus_assert:has_no_stanzas(Alice2)
        end).
```

Think of it as a DSL

```
message_routing(Config) ->
    escalus:story(Config, [2, 2], fun(Alice1, Alice2, Bob1, Bob2) ->
        % Alice sends a chat message to Bob's bare JID
        escalus:send(Alice1, escalus_stanza:chat_to(bob, <<"Coffee?">>),

        % Bob gets the message on both resources
        escalus:assert(is_chat_message, [<<"Coffee?">>],
                       escalus:wait_for_stanza(Bob1)),
        escalus:assert(is_chat_message, [<<"Coffee?">>],
                       escalus:wait_for_stanza(Bob2)),

        % Bob replies to one of Alice's resources
        escalus:send(Bob1, escalus_stanza:chat_to(Alice1, <<"Sure!">>),

        % Alice receives the reply only on that resource
        escalus:assert(is_chat_message, [<<"Sure!">>],
                       escalus:wait_for_stanza(Alice1)),
        escalus_assert:has_no_stanzas(Alice2)
    end).
```

Common Test config file

```
{escalus_users, [
    {alice, [
        {username, <<"alice">>},
        {server, <<"localhost">>},
        {password, <<"makota">>}]},
    {bob, [
        {username, <<"bob">>},
        {server, <<"localhost">>},
        {password, <<"mapsa">>}]}]}.
```

test results

- bad one
- good one

What is done for us

- User registration & de-registration.
- User login & logout.
- XML parsing & generation.

What is made easier

- Checking assertions.
- Debugging.

Lesson learned

The Good:

- acceptance testing,
- ease of writing readable tests,
- stories!

Lessons learned

The Bad: race conditions

Lessons learned

The Ugly:

- exmpp,
- XML,
- Common Test (a little).

Short story of internals rewrite

exmpp $\rightarrow$ exml + lxmppc

- internal rewrite
- type madness: atoms, strings, binaries
- warn & go

Future

- BOSH support
- property-based testing

Summary

- Think what to test.
- Make writing tests easier.
- Use Escalus for XMPP-related stuff.

That's it!

Thank you for your attention!

krzysztof.goj@erlang-solutions.com
`https://github.com/goj/escalus`