# ejabberd

Extending & Using

mila®

Hi

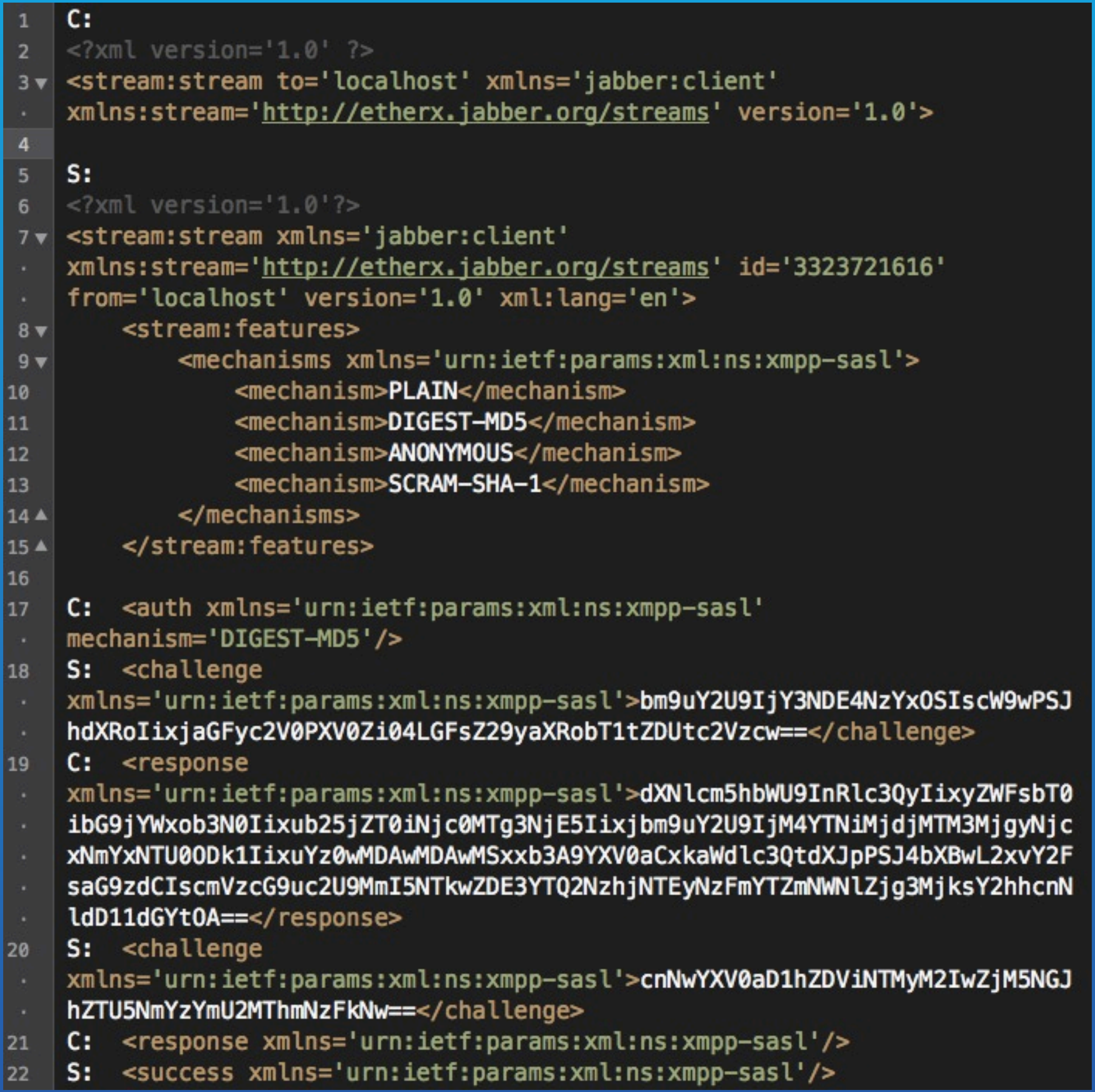
@weibelm
@mweibel

mila®

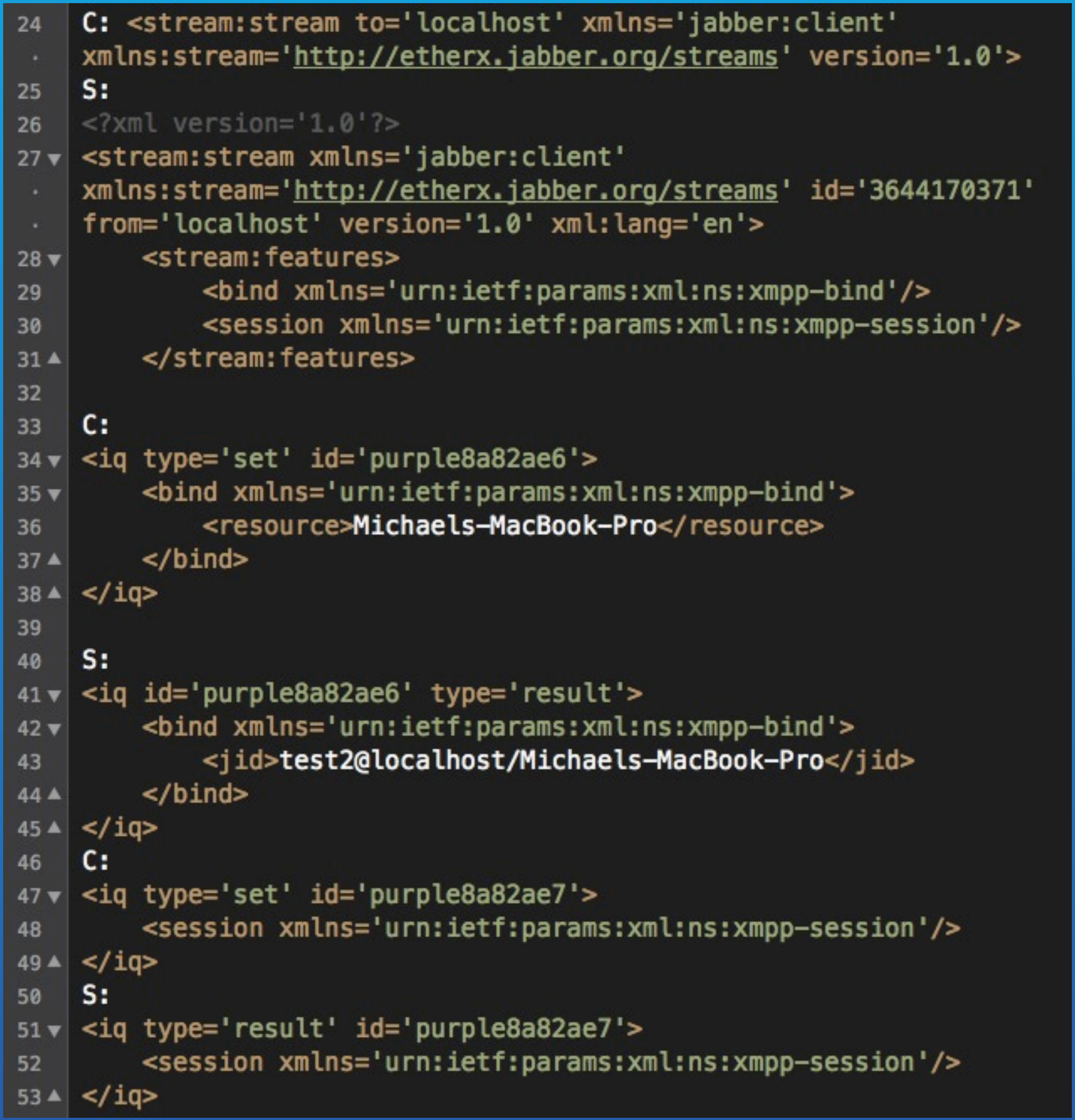
## Contents

- XMPP
- Ejabberd Basics
- Module Development

## XMPP



```
C:
<?xml version='1.0' ?>
<stream:stream to='localhost' xmlns='jabber:client'
xmlns:stream='http://etherx.jabber.org/streams' version='1.0'>

S:
<?xml version='1.0'?>
<stream:stream xmlns='jabber:client'
xmlns:stream='http://etherx.jabber.org/streams' id='3323721616'
from='localhost' version='1.0' xml:lang='en'>
    <stream:features>
        <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
            <mechanism>PLAIN</mechanism>
            <mechanism>DIGEST-MD5</mechanism>
            <mechanism>ANONYMOUS</mechanism>
            <mechanism>SCRAM-SHA-1</mechanism>
        </mechanisms>
    </stream:features>

C:  <auth xmlns='urn:ietf:params:xml:ns:xmpp-sasl'
mechanism='DIGEST-MD5'/>
S:  <challenge
xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>bm9uY2U9IjY3NDE4NzYxOSIscW9wPSJ
hdXRoIixjaGFyc2V0PXV0Zi04LGFsZ29yaXRobT1tZDUtc2Vzcw==</challenge>
C:  <response
xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>dXNlcm5hbWU9InRlc3QyIixyZWFsbT0
ibG9jYWxob3N0Iixub25jZT0iNjc0MTg3NjE5Iixjbm9uY2U9IjM4YTNiMjdjMTM3MjgyNjc
xNmYxNTU0ODk1IixuYz0wMDAwMDAwMSxxb3A9YXV0aCxkaWdlc3QtdXJpPSJ4bXBwL2xvY2F
saG9zdCIscmVzcG9uc2U9MmI5NTkwZDE3YTQ2NzhjNTEyNzFmYTZmNWNlZjg3MjksY2hhcnN
ldD11dGYtOA==</response>
S:  <challenge
xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>cnNwYXV0aD1hZDViNTMyM2IwZjM5NGJ
hZTU5NmYzYmU2MThmNzFkNw==</challenge>
C:  <response xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
S:  <success xmlns='urn:ietf:params:xml:ns:xmpp-sasl'/>
```



```
C: <stream:stream to='localhost' xmlns='jabber:client'
xmlns:stream='http://etherx.jabber.org/streams' version='1.0'>
S:
<?xml version='1.0'?>
<stream:stream xmlns='jabber:client'
xmlns:stream='http://etherx.jabber.org/streams' id='3644170371'
from='localhost' version='1.0' xml:lang='en'>
    <stream:features>
        <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'/>
        <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
    </stream:features>

C:
<iq type='set' id='purple8a82ae6'>
    <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
        <resource>Michaels-MacBook-Pro</resource>
    </bind>
</iq>

S:
<iq id='purple8a82ae6' type='result'>
    <bind xmlns='urn:ietf:params:xml:ns:xmpp-bind'>
        <jid>test2@localhost/Michaels-MacBook-Pro</jid>
    </bind>
</iq>
C:
<iq type='set' id='purple8a82ae7'>
    <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
</iq>
S:
<iq type='result' id='purple8a82ae7'>
    <session xmlns='urn:ietf:params:xml:ns:xmpp-session'/>
</iq>
```

## Basics

- RFC 6120, 6121 and 6122
- XEPs for extensions

## Basics

- 1-1 Chat
- Multi-User-Chat (MUC)
- Publish-Subscribe (Pubsub)

## Basics

- Jingle
- Federation

## Stanza

- Message
- Presence
- IQ

## Web

- BOSH
- Websockets

## ejabberd

## Basics

- Open Source
- Commercial Version by ProcessOne

- www.ejabberd.im
- www.process-one.net

## Basics

- erlang R14
- ejabberd 2.1 & ejabberd 3.0

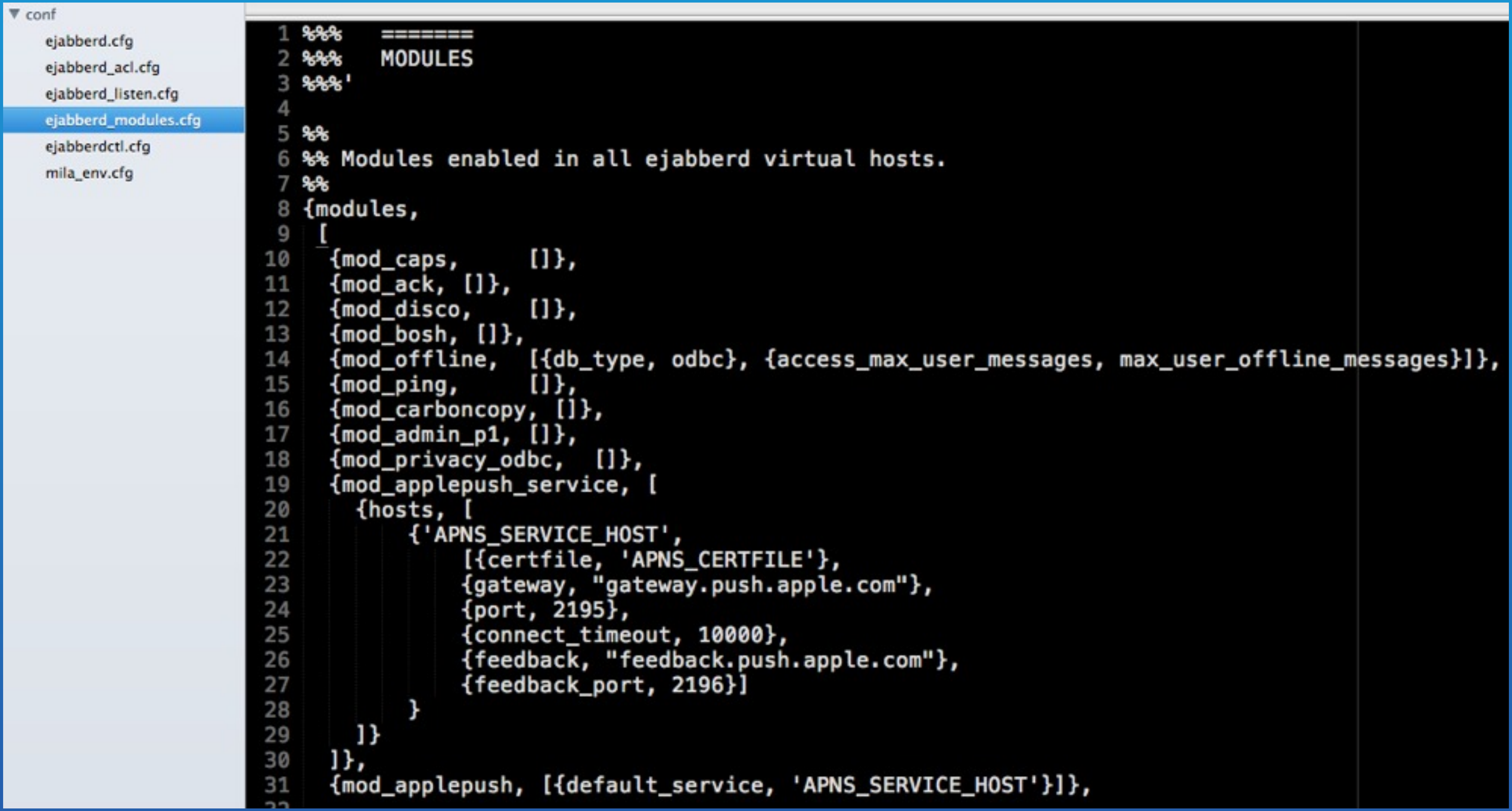
## Basics

- lots of built-in modules
- extendable

## Configuration

- ejabberd.cfg
- splittable into multiple files
- constants

# Configuration



```
%%%   =======
%%%   MODULES
%%%'

%%
%% Modules enabled in all ejabberd virtual hosts.
%%
{modules,
 [
  {mod_caps,      []},
  {mod_ack, []},
  {mod_disco,     []},
  {mod_bosh, []},
  {mod_offline,   [{db_type, odbc}, {access_max_user_messages, max_user_offline_messages}]},
  {mod_ping,      []},
  {mod_carboncopy, []},
  {mod_admin_p1, []},
  {mod_privacy_odbc,   []},
  {mod_applepush_service, [
    {hosts, [
        {'APNS_SERVICE_HOST',
            [{certfile, 'APNS_CERTFILE'},
            {gateway, "gateway.push.apple.com"},
            {port, 2195},
            {connect_timeout, 10000},
            {feedback, "feedback.push.apple.com"},
            {feedback_port, 2196}]
        }
    ]}
  ]},
  {mod_applepush, [{default_service, 'APNS_SERVICE_HOST'}]},
```

## Module Development

## Auth Modules

## Core Auth Modules

- internal
- anonymous
- external
- odbc
- pam
- ldap

## Own Authentication

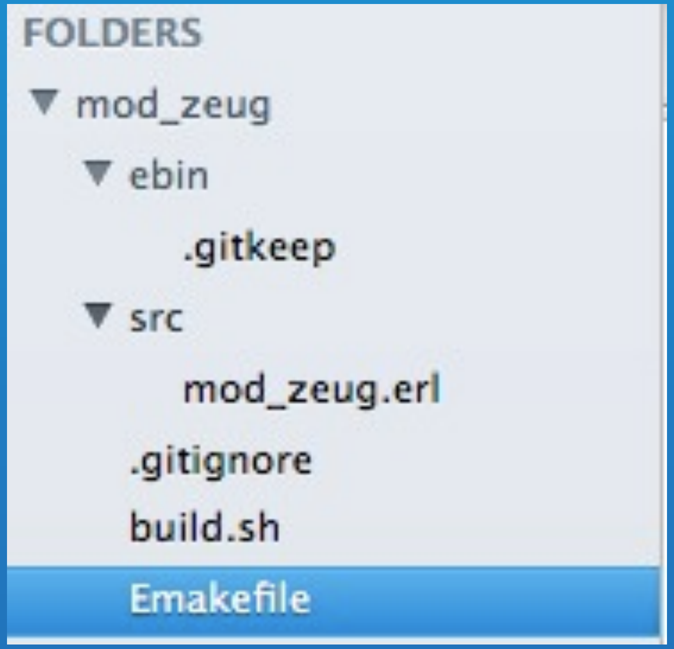
- implement ejabberd_auth
  - check_password/3
  - store_type/0
  - plain_password_required/0
  - is_user_exists/2
  - etc.

## No custom mechanism :(

## Stanza interceptors

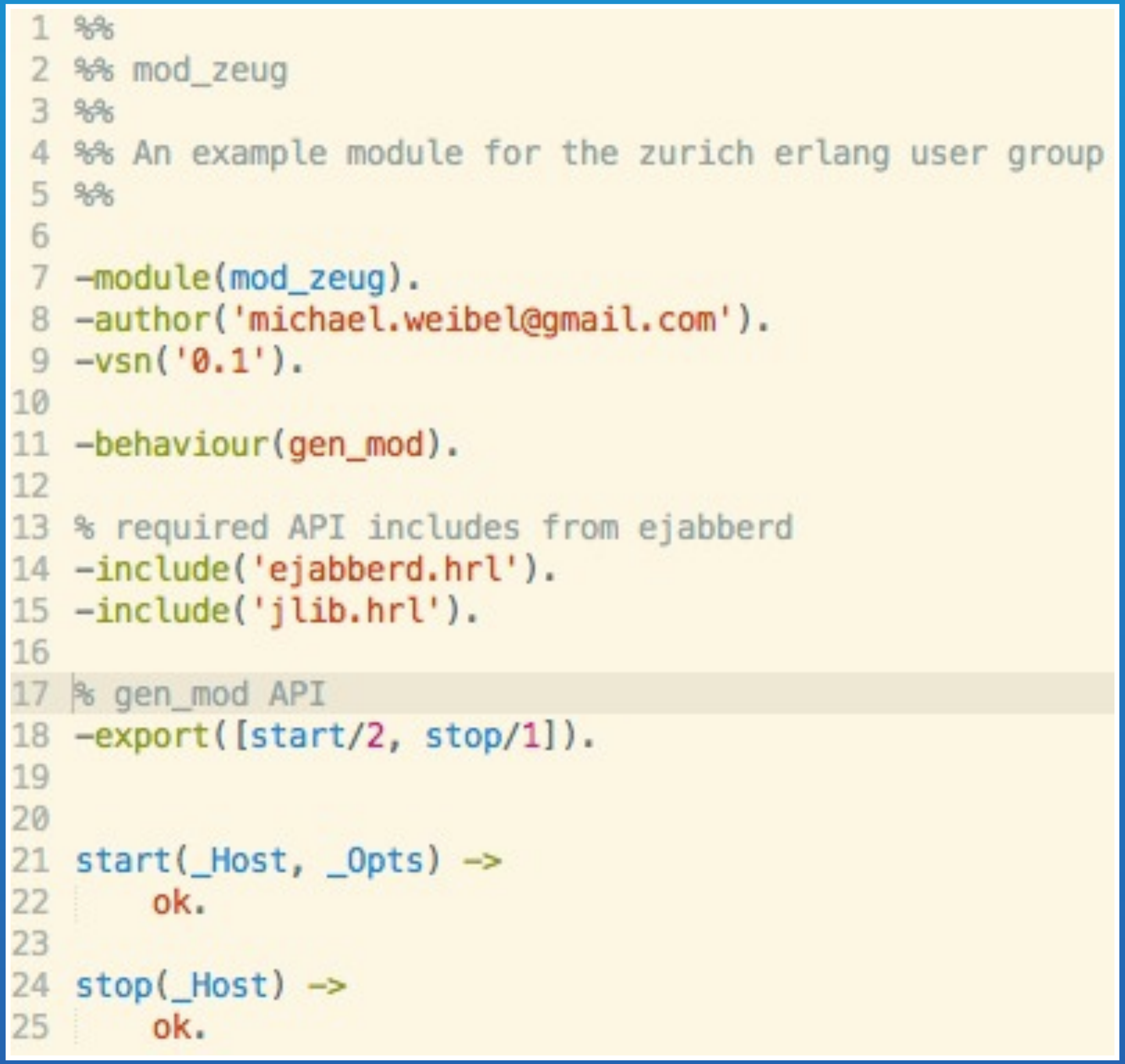
## Let‘s write an own Module

## Setup


FOLDERS
▼ mod_zeug
▼ ebin
.gitkeep
▼ src
mod_zeug.erl
.gitignore
build.sh
Emakefile

# gen_mod

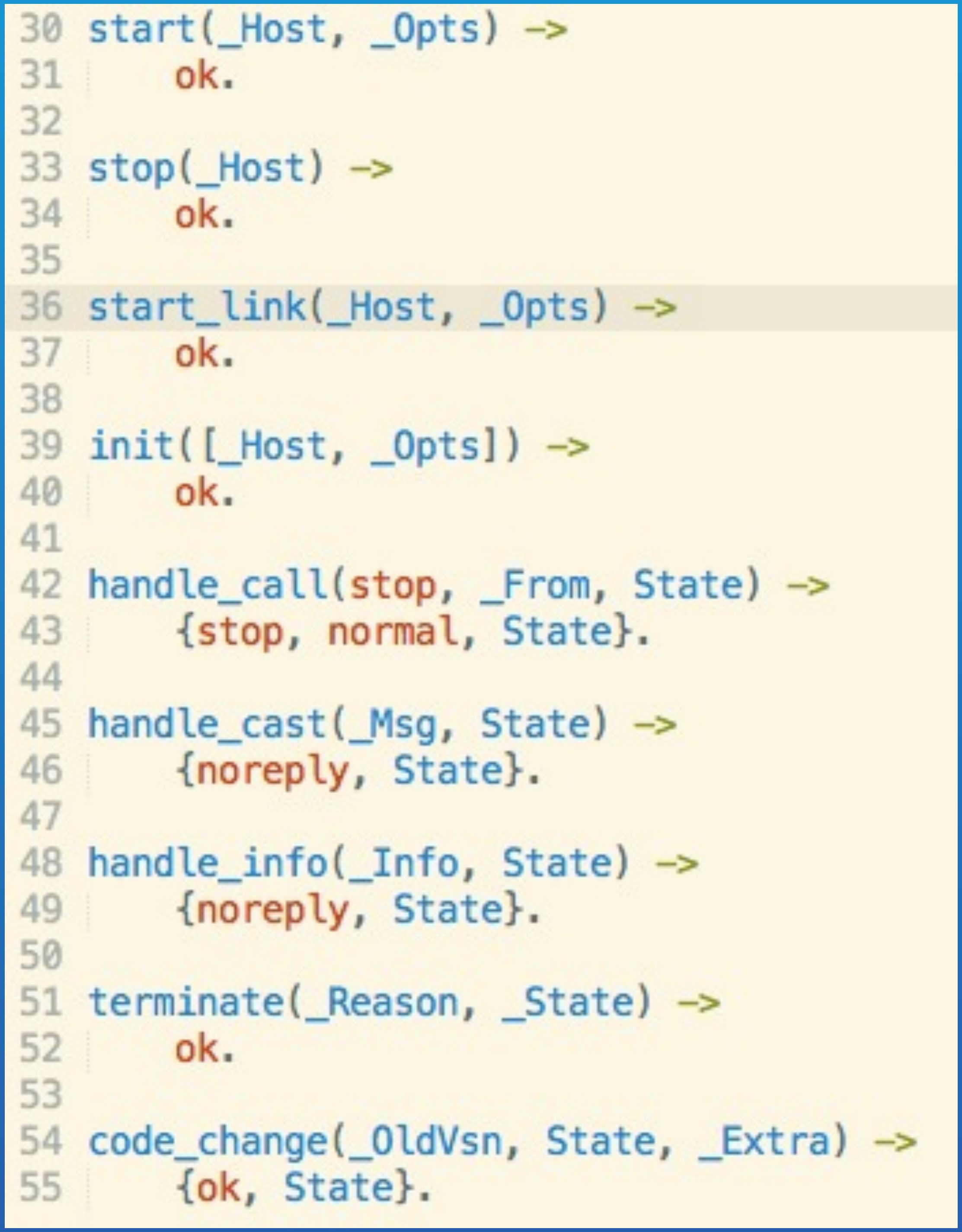
- start/2
- stop/1



```
%%
%% mod_zeug
%%
%% An example module for the zurich erlang user group
%%

-module(mod_zeug).
-author('michael.weibel@gmail.com').
-vsn('0.1').

-behaviour(gen_mod).

% required API includes from ejabberd
-include('ejabberd.hrl').
-include('jlib.hrl').

% gen_mod API
-export([start/2, stop/1]).


start(_Host, _Opts) ->
    ok.

stop(_Host) ->
    ok.
```

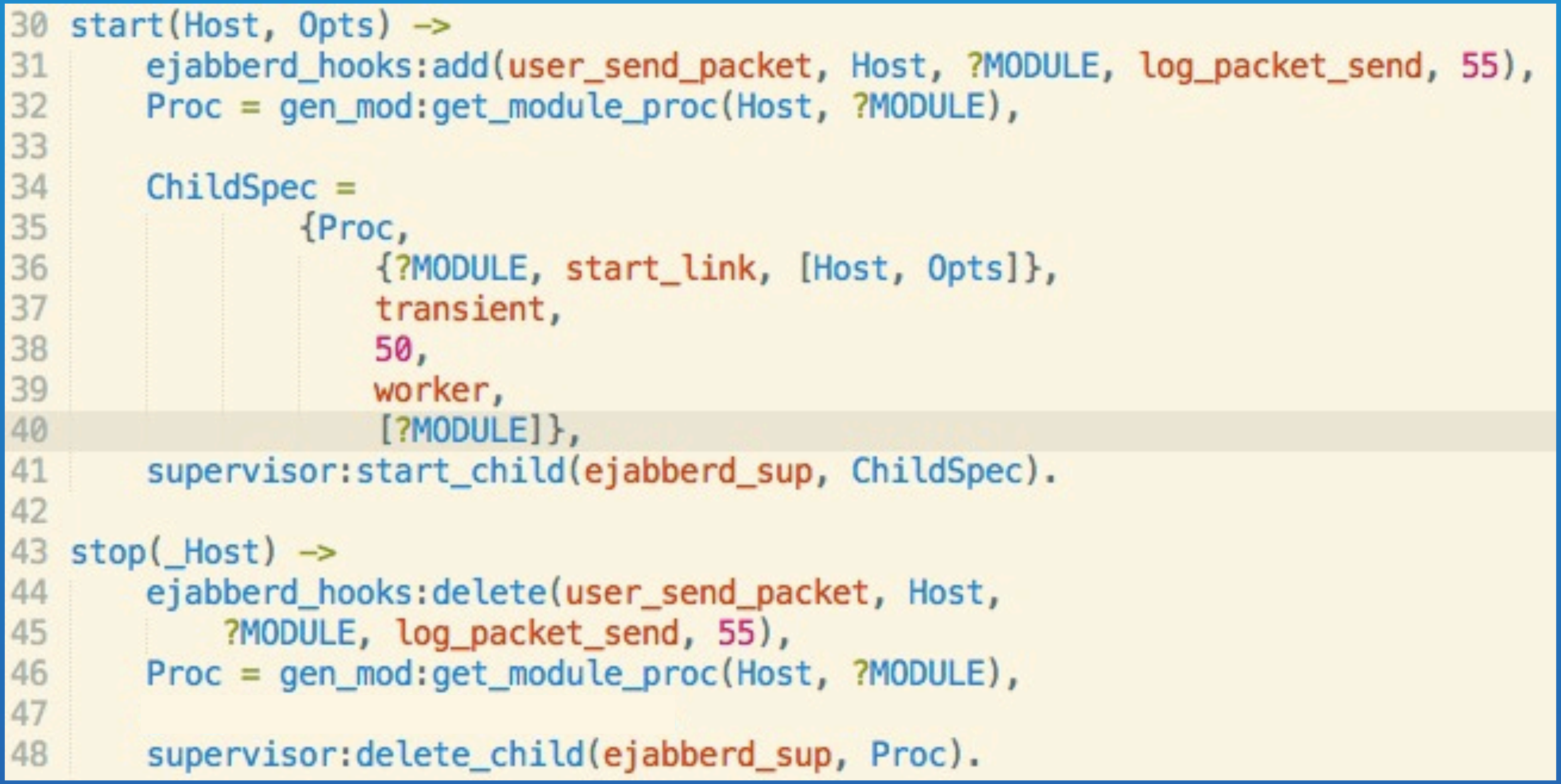
# gen_server

- Just for fun :)



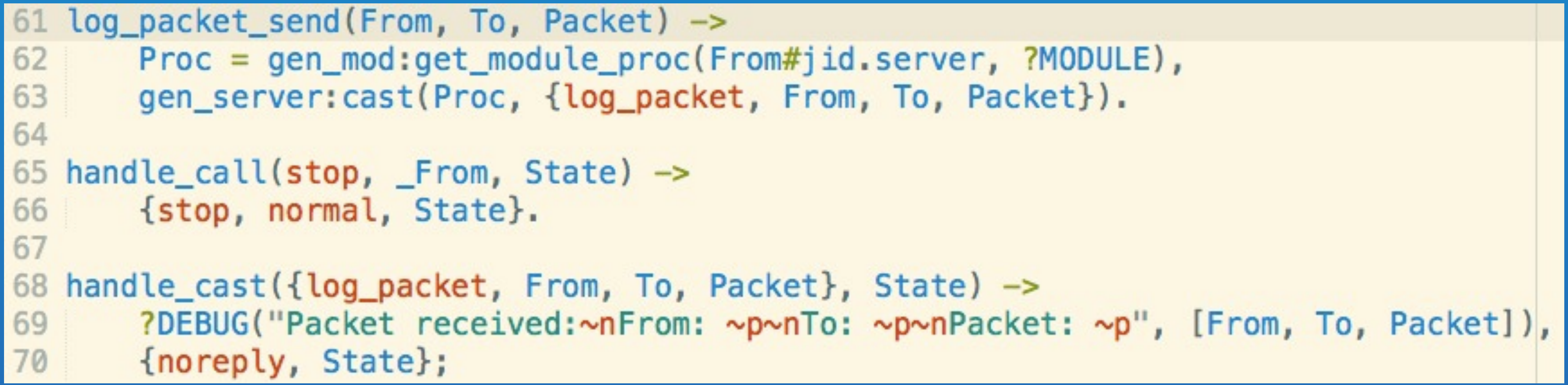
```erlang
start(_Host, _Opts) ->
    ok.

stop(_Host) ->
    ok.

start_link(_Host, _Opts) ->
    ok.

init([_Host, _Opts]) ->
    ok.

handle_call(stop, _From, State) ->
    {stop, normal, State}.

handle_cast(_Msg, State) ->
    {noreply, State}.

handle_info(_Info, State) ->
    {noreply, State}.

terminate(_Reason, _State) ->
    ok.

code_change(_OldVsn, State, _Extra) ->
    {ok, State}.
```

# Setup hook & gen_server



```erlang
start(Host, Opts) ->
    ejabberd_hooks:add(user_send_packet, Host, ?MODULE, log_packet_send, 55),
    Proc = gen_mod:get_module_proc(Host, ?MODULE),

    ChildSpec =
            {Proc,
                {?MODULE, start_link, [Host, Opts]},
                transient,
                50,
                worker,
                [?MODULE]},
    supervisor:start_child(ejabberd_sup, ChildSpec).

stop(_Host) ->
    ejabberd_hooks:delete(user_send_packet, Host,
        ?MODULE, log_packet_send, 55),
    Proc = gen_mod:get_module_proc(Host, ?MODULE),

    supervisor:delete_child(ejabberd_sup, Proc).
```

# Hook listener



```erlang
log_packet_send(From, To, Packet) ->
    Proc = gen_mod:get_module_proc(From#jid.server, ?MODULE),
    gen_server:cast(Proc, {log_packet, From, To, Packet}).

handle_call(stop, _From, State) ->
    {stop, normal, State}.

handle_cast({log_packet, From, To, Packet}, State) ->
    ?DEBUG("Packet received:~nFrom: ~p~nTo: ~p~nPacket: ~p", [From, To, Packet]),
    {noreply, State};
```

## Upsides

## Erlang

## ProcessOne

## Downsides

## Documentation

## Community

## No rebar (yet)

## No mobile reliability (by default)

No websockets
(by default)

## No stats builtin

## Tools

## ej.sh

- ej build
- ej tail
- ej conf
- ej restart
- all ejabberdctl commands

https://gist.github.com/pstadler/3918130

## Alternatives

- MongooseIM (ejabberd fork)
- Tigase (java)
- Prosody (lua)
- Openfire (java)

Thanks.

Questions?