Erlang Academy

Лекция 6

Автор курса: Костюшкин Сергей

План

- Очередь, Стек, Куча
- Паралеллизм
- Процессы
- Оператор recieve
- Типы связей между процессами
- Поведения (behaviours)
- Поведения встроенные в OTP
- Фреймворк для тестирования Common Test

Очередь

Абстрактная коллекция данных, работа с которой ведется в точном соответствии с порядком добавления элементов.
FIFO (First-In-First-Out)

Стек

Абстрактная коллекция данных, работа с которой ведется в порядке обратном порядку добавления элементов.
LIFO (Last-In-First-Out)

Erlang-список организован как стек

Куча

Абстрактная структура данных организованая как очередь с приоритетом. Также способ организации хранения данных в процессах

Процесс

Процесс Erlang следует рассматривать, как комбинацию стека, кучи и очереди хранящей сообщения (почтовый ящик)

Паралеллизм

Виды параллелизма:

- Паралельность — взаимодействие паралельных процессов
- Конкурентность — взаимодействие последовательных процессов

Конкурентность базируется на обработке прерываний

Процессы

```
1> X = 99.
99
2> F = fun() -> X + 1 end.
#Fun<erl_eval.20.50752066>
3> Pid = spawn(F).
<0.37.0>
```

Процессы

```
1> F = fun() ->
1>     receive {ping, Pid} -> Pid ! pong end
1> end.
#Fun<erl_eval.20.50752066>
2> Pid = spawn(F).
<0.36.0>
3> Pid ! {ping, self()}.
{ping,<0.33.0>}
4> receive
4>     Msg -> io:format("Receive msg:~p~n", [Msg])
4> end.
Receive msg:pong
ok
```

Рекурсивное чтение сообщений

```
process_loop() ->
    receive
        {ping, Pid} ->
            Pid ! pong;
    end,
    process_loop().
```

Типы связей между процессами

- monitor
- link

Поведения встроенные в OTP

gen_server

supervisor

application

gen_statem (Заменил gen_fsm)

gen_event

OTP Design Principles

OTP Design Principles

Common Test

Common Test User's Guide