Erlang Academy

Лекция 5

Автор курса: Костюшкин Сергей

План

- Метапрограммирование
- Параметризированые модули
- Списки свойств (модули lists и proplists)
- Записи (records)
- ETS
- DETS
- MatchSpec - Язык запросов в ETS и DETS

Метапрограммирование

Парадигма программирования при которой программы трактуются как данные. Это означает что программа может читать, гененерировать и изменять другие программы или даже себя.

Виды метапрограммирования:

- Кодогенерация
- Статический анализ кода
- DSL (Domain Specific language) — предметно-специфичный язык

Параметризированые модули

Модуль:

```
-module(param_example, [Name, Age]).
-export([name/0]).

name() -> Name.
```

Консоль:

```
1> Obj = param_example:new("Tony", 18).
2> Obj:name().
"Tony"
```

proplists

```
1> Proplist = [{name, "Santa"}, {age, 1054}].
[{name,"Santa"},{age,1054}]
2> proplists:get_value(name, Proplist).
"Santa"
```

maps

```
1> Map = #{name => "Santa", age => 1054}.
#{age => 1054,name => "Santa"}
2> maps:get(name, Map).
"Santa"
3> #{age := Age} = Map.
#{age => 1054,name => "Santa"}
4> Age.
1054
5> Map2 = Map#{age => 2000}.
#{age => 2000,name => "Santa"}
```

Process Dictionary

```
1> put(key1, value1).
undefined
2> put(key2, value2).
undefined
3> put(key3, value3).
undefined
4> get().
[{key3,value3},{key2,value2},{key1,value1}]
5> get(key2).
value2
6> get_keys().
[key3,key2,key1]
7> erase().
[{key3,value3},{key2,value2},{key1,value1}]
```

dict

```
1> D = dict:new().
{dict,0,16,16,8,80,48,
      {[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[]},
      {{[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[]}}}
2> D2 = dict:append(key1, value1, D).
{dict,1,16,16,8,80,48,
      {[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[]},
      {{[],
        [[key1,value1]],
        [],[],[],[],[],[],[],[],[],[],[],[],[],[]}}}
3> dict:size(D2).
1
4> dict:erase(key1, D2).
{dict,0,16,16,8,80,48,
      {[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[]},
      {{[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[]}}}
```

Записи (records)

```
-module(records_demo).
-export([new/0, new/3]).
-export([get_field/2, set_field/3, get_index/1]).

-record(person, {name=”Joe”, gender, age=56}).

new() ->
    #person{}.

new(Name, Gender, Age) ->
    #person{name=Name, gender=Gender, age=Age}.
```

Записи (records) продолжение

```
get_index(name)   -> #person.name;
get_index(gender) -> #person.gender;
get_index(age)    -> #person.age.

get_field(name, Record)   -> Record#person.name;
get_field(gender, Record) -> Record#person.gender;
get_field(age, Record)    -> Record#person.age.

set_field(name, Name, Record) ->
    Record#person{name=Name};
set_field(age, Age, Record) ->
    Record#person{age=Age}.
```

Записи (records)

```
1> records_demo:new().
{person,"Joe",undefined,56}
2> records_demo:new("Serhii", male, 33).
{person,"Serhii",male,33}
3> Person1 = records_demo:new().
{person,"Joe",undefined,56}
4> Person2 = records_demo:new("Serhii", male, 33).
{person,"Serhii",male,33}
5> records_demo:get_index(name).
2
6> records_demo:get_index(gender).
3
7> records_demo:get_index(age).
4
8> records_demo:get_field(name, Person1).
"Joe"
9> Person2New = records_demo:set_field(name, "Serhii Kostiushkin", Person2).
{person,"Serhii Kostiushkin",male,33}
```

ETS (Erlang Term Storage)

```
EtsId = ets:new(TableName, Opts).
true = ets:insert(EtsId, ObjectOrObjects).
[Object] = ets:lookup(EtsId, Key).
true = ets:delete(EtsId).
```

ETS (Erlang Term Storage)

```
1> ets:new(persons, [public, named_table]).
persons
2> ets:insert(persons, {key, value1, value2, value3}).
true
2> ets:insert(persons, {another_key, value}).
true
3> ets:lookup(persons, key).
[{key,value1,value2,value3}]
4> ets:delete(persons, key).
true
5> ets:delete(persons).
true
```

DETS (Disk Erlang Term Storage)

```
{ok, Name} = dets:open_file(Name, Opts).
ok = dets:insert(Name, ObjectOrObjects).
[Object] = dets:lookup(Name, Key).
ok = dets:close(Name).
```

DETS (Disk Erlang Term Storage)

```
1> dets:open_file(persons, []).
{ok,persons}
2> dets:insert(persons, {key1, value1}).
ok
3> dets:lookup(persons, key1).
[{key1,value1}]
4> dets:close(persons).
ok
```

MatchSpec

```
1> MatchHead = {'$1','$2'}.
{'$1','$2'}
2> Guard = [{'>','$2',3}].
[{'>','$2',3}]
3> Result = ['$1'].
4> MatchFunction = {MatchHead, Guard, [Result]}.
{{'$1','$2'},[{'>','$2',3}],[['$1']]}
5> MatchSpec = [MatchFunction].
[{{'$1','$2'},[{'>','$2',3}],[['$1']]}]
6> MatchSpec2 = [{{'$1','$2'},[{'>','$2',3}],['$$']}].
[{{'$1','$2'},[{'>','$2',3}],['$$']}]
```

MatchSpec

```
1> ets:new(match_table, [named_table, public, set]).
match_table
2> ets:insert(match_table, {boy, 14}).
true
3> ets:insert(match_table, {girl, 1}).
true
4> ets:select(match_table, [{{'$1','$2'},[{'>','$2',3}],[['$1']]}]).
[[boy]]
5> MatchSpec2 = [{{'$1','$2'},[{'>','$2',3}],['$$']}].
[{{'$1','$2'},[{'>','$2',3}],['$$']}]
6> ets:select(match_table, MatchSpec2).
[[boy,14]]
```

Что почитать?

Документация по Process Dictionary

Документация по устройству записей

Документация по ETS

Документация по устройству MatchSpec