Industrial Functional Programming [1]

Melinda Tóth, István Bozó

Dept. Programming Languages and Compilers
Eötvös Loránd University, Budapest, Hungary

[1] Supported by TÁMOP-4.1.2.A/1-11/1-2011-0052

Contents

Erlang VM

- The Prolog interpreter – 1986
- JAM - Joe’s Abstract Machine – 1989
- BEAM - (Bogdan’s) Björn’s Erlang Abstract Machine

Erlang Evaluation

- Erlang emulator
- Erlang code compiled to bytecode
- Loaded to the VM
- Evaluated by the VM
- Elang shell: read-eval-print loop

Erlang shell

- `erl, erl.exe, werl.exe`
- `1 + 2.  "apple".`
- `q().  init:stop().`
- BREAK menu: `Ctrl-C / Ctrl-Break`
- User Switch Command: `Ctrl-G` – job contorller

Useful Shell Commands

- `help()`
- `h()`
- `i()`
- `memory()`
- `c(ModName)`
- `ls(), ls(Dir)`
- `b()`
- `f(), f(X)`
- `e(Number),e(-1)`
- `v()`
- `module_name:function_name(Params)`
- `m(ModName), module_name:module_info()`
- `pwd(), cd(Path)`

Types – Terms

- Numbers (Integer, Float)
- Binaries/Bitstrings
- Atoms
- Tuples
- Lists (Strings)
- Unique identifiers: pids, ports, references
- Funs

Numbers

- Integer – `10, 2#10101, 36#PQ3, $w`
- Floats – `0.01, 17.2, 11.12E-10`
- Arithmetic operators: `+, -, *, /, div, rem`
- `math` module -> `sqrt, pow`
- `N bsl K, N bsr K`
- `band, bor, bxor, bnot`

Binaries & Bitstrings

- Binary: sequence of bytes
- Bitstring: sequence of bit
- `<<>>, << 0,1,2,3>>`
- `<< "hello", 0, "dummy">>`

Atoms

- String constant
- Atom – lower + letter + digit + @ + _
- `'An atom', an_atom1`
- %% Boolean – `true, false`
- `ok & undefined`
- Maximum length: 255 characters
- Maximum number of atoms 1048576 (default)

Tuple

- Fixed number of elements (n-tuple)
- Tuple – `{...}`
- `{foo, bar, {1,2}, "gazonk"}`
- `{}`
- Tagged tuple: `{int, 42}`, `{pos, 23, 43}`
- `element(n, Tuple)`

List

- Data structure with dynamic length
- List – `[...]`, String – `"..."`
- `[], "", [1,23, {foo, 2}, [bar,  [gazonk]]]`
- `[1 | []], [1 | [2]], [1,2,3 | [4,5]]`
- `[1,2] ++ [3,4,5]`
- `[97,98,99,100] == "abcd" == [$a,$b,$c,$d]`
- ??? `[0 | v(Num)]` ???
- Proper & improper lists

Unique identifiers and Funs

- Pid – `< 0.4.2>`
- Port – `#Port<0.472>`
- Reference – `#Ref<0.0.0.42>`
- Fun – `#Fun<...>`

Terms

- Integer – `10` (binary, octal, hexadecimal etc.)
- Floats – `17.2, 11.12E-10`
- Binaries and Bitstrings
- Atom – lower + letter + digit + @ + _
- %% Boolean – `true, false`
- Tuple – `{...}`
- List – `[...]` , String – `"..."`
- Unique identifiers: pid, port, reference
- Fun

Comparison Of Types&Terms

number < atom < fun < port < pid < tuple < list < binary

<	>
=<	>=
/=	==
=:=	=/=

On the Next Lecture ...

- Variables
- Pattern Matching
- Modules
- Functions