


Langkah-langkah Menjalankan Erlang untuk Pertama Kali

Laporan PKL Kedua – Ayu Ramadhan Mulia

Sebelumnya kita telah sukses menginstal Erlang pada Ubuntu melalui *terminal*. Selanjutnya, untuk memudahkan user membuat program dan meng-*compile*-nya, Erlang biasanya disupport oleh text editor, Emacs. Untuk panduan menginstal Emacs pada Ubuntu dapat mengikuti langkah-langkah berikut:

1. Jalankan terminal (Application → Accessories → Terminal)
2. Masuk ke root mode (ketik `sudo bash` → masukkan password root)
3. Pada root mode, ketik “`sudo apt-get install emacs`”, maka System kemudian akan mencari ke repository semua paket Emacs untuk kemudian ditanamkan ke system komputer kita.
4. Setelah selesai terinstal, masih di terminal dalam keadaan root mode, ketik “`nano ~/.emacs`”, maka akan muncul kotak dialog kosong, kemudian isi dengan data berikut:



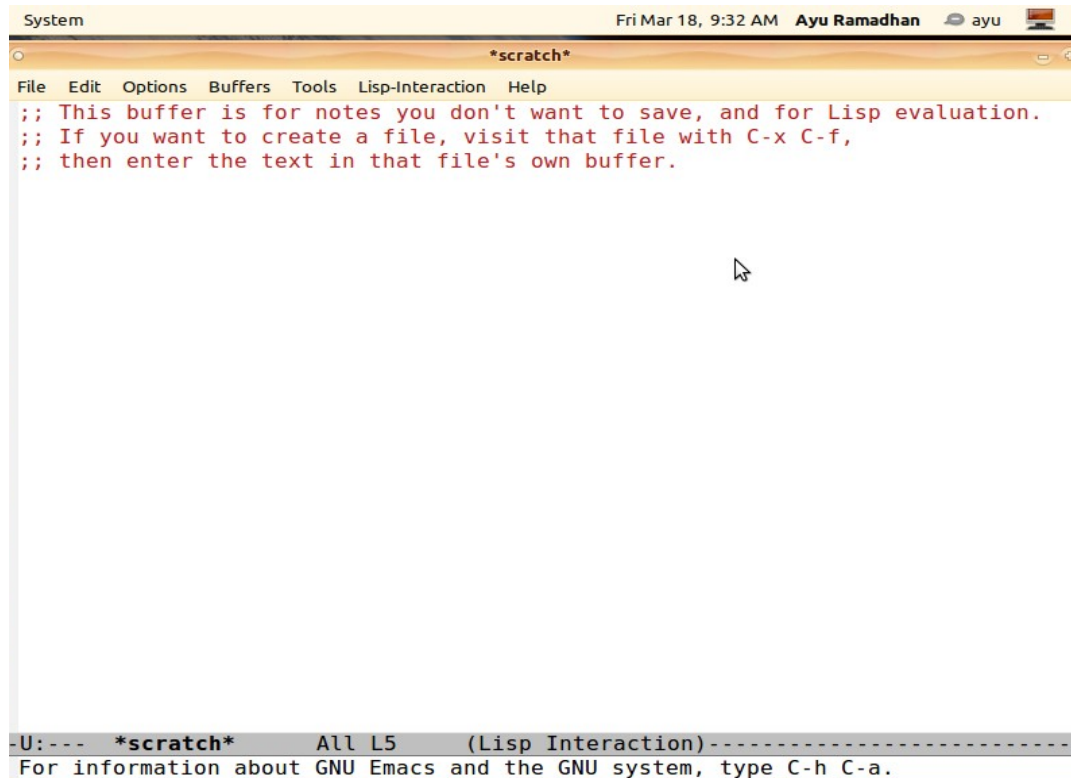
```
root@ayu-laptop: ~
File Edit View Search Terminal Help
GNU nano 2.2.4 File: /home/ayu/.emacs

;; disable bell function
(setq ring-bell-function 'ignore)
;;disable toolbar
(tool-bar-mode -1)
;;disable scrollbar
(toggle-scrollbar -1)
;;disable splash screen
(custom-set-variables '(inhibit-startup-screen t))
;; current buffer name in title bar
(setq frame-title-format "%b")
;; starts emacs server
(server-start)
```

[Read 12 lines]

^G Get Help **^O** WriteOut **^R** Read File **^Y** Prev Page **^K** Cut Text **^C** Cur Pos

data tersebut berfungsi untuk membuat tampilan Emacs menjadi seperti berikut:



Dengan begitu Emacs sudah bisa dijalankan sebagai text editor untuk membuat module-module yang nantinya dapat digunakan dalam membuat program berbasis Erlang.

Menggunakan Emacs untuk Membuat Module Sederhana

Setelah terinstall dengan sukses Emacs kemudian sudah dapat digunakan untuk membuat module-module yang kemudian dapat dikompilasi menjadi sebuah program. Untuk menjalankan Emacs cukup dengan mengklik Application → Accessories → GNU Emacs 23 atau Application → Programming → GNU Emacs 23.

setelah text editor Emacs terbuka kita sudah bisa mulai dengan membuat modul-modul sederhana. Sebelum membuat modul alangkah baiknya kita mengenal elemen-elemen apa saja yang biasanya ada di dalam bahasa pemrograman Erlang. Berikut penjabarannya melalui contoh-contoh modul:

Modul yang diberi nama “formula” dan “formula2”:

```
-module(formula).                                -module(Formula2).

-export([convert_length/1]).                    -export([list_length/1]).

convert_length({centimeter, X}) ->           list_length([]) ->
{inch, X / 2.54};                               0;

                                                list_length([First | Rest]) ->
                                                1 + list_length(Rest).
```

Penjelasan:

- ****** : nama module
- **☐** : modul formula memiliki sebuah fungsi yang disebut convert_length yang berisikan satu argumen (X).
- **☐** : tuples (sejenis struct di bahasa C, menggunakan kurung kurawal “{ }”)
→ centimeter : atom
→ X : Variable
- **☐** : List (merepresentasikan daftar dari suatu kumpulan elemen

contoh: [First | TheRest] = [1,2,3,4,5,].) List ditandai frmg

Setelah mengerti elemen-elemen yang nantinya akan selalu kita gunakan untuk mengembangkan program berbasis Erlang, sekarang kita akan mencoba membuat modul sederhana yang dinamakan “formula.erl” dengan isi sebagai berikut.

```

formula.erl
File Edit Options Buffers Tools Erlang Help
%%;; This buffer is for notes you don't want to save, and for Lisp evaluation.
%%;; If you want to create a file, visit that file with C-x C-f,
%%;; then enter the text in that file's own buffer.

-module(formula).
-export([volume/1, area/1, glbb/1]).
volume({kubus, Sisi}) -> Sisi * Sisi * Sisi;
volume({tabung, Radius, Tinggi}) ->
    3.14159 * Radius * Radius * Tinggi.

area({kubus, Sisi}) ->
    Sisi * Sisi;
area({balok, Panjang, Lebar}) ->
    Panjang * Lebar;
area({segitiga, Alas, Tinggi}) ->
    (1 / 2) * Alas * Tinggi.

glbb({speed, Distance, Time}) ->
    {speed, Distance / Time};
glbb({distance, Speed, Time}) ->
    {distance, Speed * Time};
glbb({time, Speed, Distance}) ->
    {time, Distance / Speed}.

```

Berikut penjabarannya:

```

-module(formula).
-export([volume/1, area/2, glbb/2]).

volume({kubus, Sisi}) -> Sisi * Sisi * Sisi;

area({balok, Panjang, Lebar}) -> Panjang * Lebar;
area({segitiga, Alas, Tinggi}) -> (1 / 2) * Alas * Tinggi.

glbb({speed, Distance, Time}) ->
    {speed, Distance / Time};
glbb({distance, Speed, Time}) ->{distance, Speed * Time};
glbb({time, Speed, Distance}) ->{time, Distance / Speed}.

```

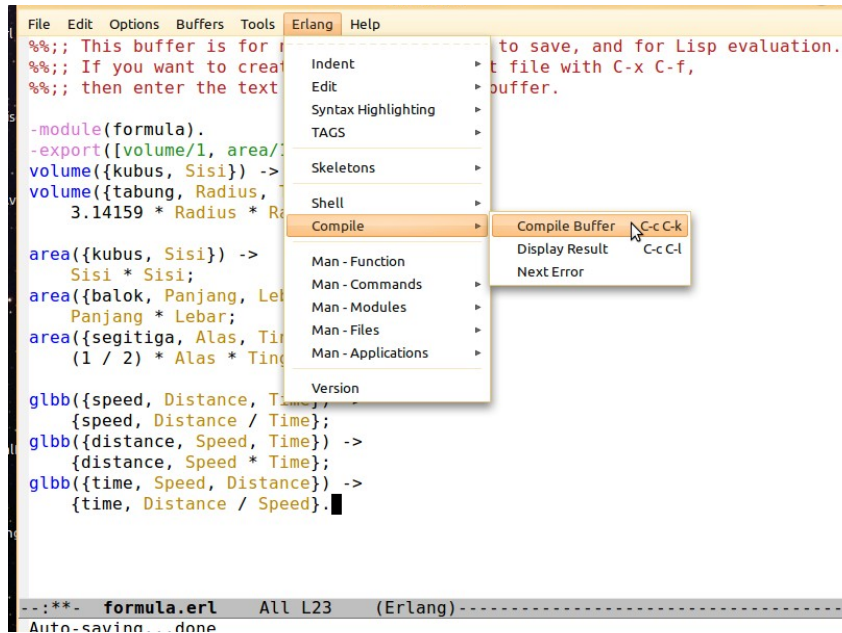
**** di dalam modul formula ada tiga fungsi yang akan dijalankan, yaitu volume dengan satu argumen (volume/1 → Sisi), area dengan dua argumen (area/2 → Panjang, Lebar), glbb dengan 2 argumen (glbb/2 → Speed, Time).**

**** Disisi sebelah kanan adalah rumus yang akan dijalankan berdasarkan nilai masukan yang diberikan. Sama halnya dengan fungsi area dan glbb(gerak lurus berubah beraturan)**

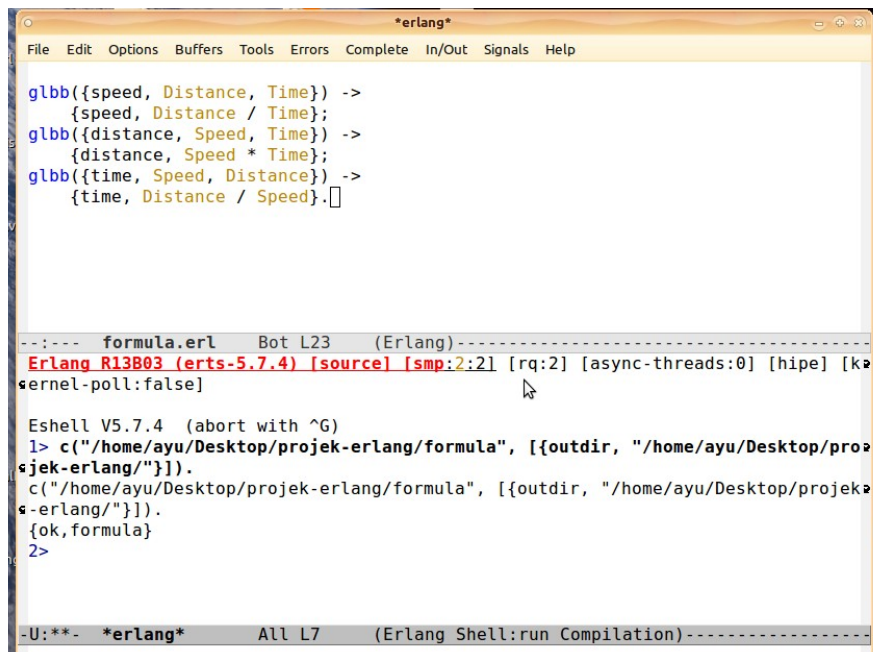
****setiap akhir fungsi diakhiri dengan tanda “.” menandakan bahwa pada fungsi tersebut sudah tidak ada lagi bagian dari fungsi lain yang ikut serta, alias fungsi berakhir di tanda titik tersebut. Sedangkan tanda “;” menandakan ada kelanjutan dari fungsi tersebut (fungsi yang sama, hanya isi tuples dan formulanya yang berbeda).**

****setelah tanda “->” itu menandakan aksi apa yang akan diberikan dengan variabel yang diketahui di dalam table, rumus aritmatikanya sama seperti rumus aritmatika yang kita kenal pada umumnya, contoh: Luas balok = Panjang * Lebar.**

Untuk menjalankan modul yang sudah kita buat, kita bisa langsung klik di tab Erlang di sisi atas jendela, kemudian klik compile → Compile Buffer, seperti gambar dibawah:



setelah di kompilasi jendela Erlang berubah menjadi:



baris pertama "1>" menunjukkan bahwa module formula ketika dicompile berjalan ok alias tidak ada error yang ditemukan. Kemudian kita dapat menjalankan apa yang sudah ditulis pada module "formula.erl" dengan melakukan pemanggilan fungsinya: "namaModule:namaFungsi...."

```

File Edit Options Buffers Tools Errors Complete In/Out Signals Help
%%;; then enter the text in that file's own buffer.

-module(formula).
-export([volume/1, area/1, glbb/1]).
volume({kubus, Sisi}) -> Sisi * Sisi * Sisi;
volume({tabung, Radius, Tinggi}) ->
    3.14159 * Radius * Radius * Tinggi.
[]

area({kubus, Sisi}) ->
    Sisi * Sisi;
area({balok, Panjang, Lebar}) ->
    Panjang * Lebar;
area({segitiga, Alas, Tinggi}) ->
    .5 * Alas * Tinggi.
-----
--:--- formula.erl 20% L10 (Erlang)-----
Erlang R13B03 (erts-5.7.4) [source] [smp:2:2] [rq:2] [async-threads:0] [hipe] [kernel-poll:false]

Eshell V5.7.4 (abort with ^G)
1> c("/home/ayu/Desktop/projek-erlang/formula", [{outdir, "/home/ayu/Desktop/projek-erlang/"}]).
c("/home/ayu/Desktop/projek-erlang/formula", [{outdir, "/home/ayu/Desktop/projek-erlang/"}]).
{ok, formula}
2> formula:volume({kubus, 5}).
125
3>
-U:*** *erlang* All L9 (Erlang Shell:run Compilation)-----

```

namaModule:namaFungsi({atoms, nilaiVariabel}).

(dari pemanggilan fungsi itu, rumus volume kubus yang akan dijalankan, maka $S * S * S = 5 * 5 * 5 = 125$)

begitulah cara pemanggilan module Formula dan fungsi-fungsi yang ada didalamnya.

Untuk mencoba mengeluarkan output tulisan kita bisa juga langsung mengetikkannya disini:

```

File Edit Options Buffers Tools Errors Complete In/Out Signals Help

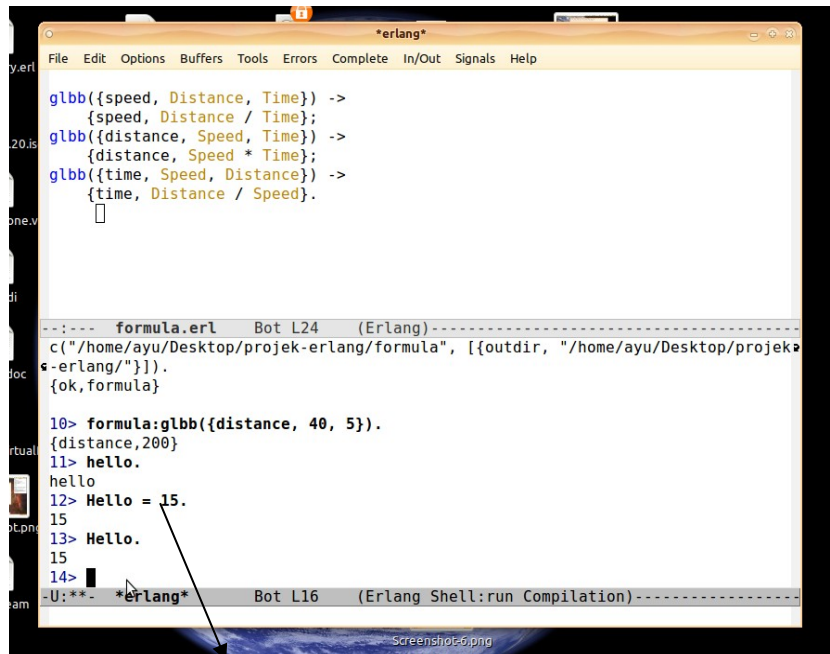
glbb({speed, Distance, Time}) ->
    {speed, Distance / Time};
glbb({distance, Speed, Time}) ->
    {distance, Speed * Time};
glbb({time, Speed, Distance}) ->
    {time, Distance / Speed}.
[]

-----
--:--- formula.erl Bot L24 (Erlang)-----
Eshell V5.7.4 (abort with ^G)
1> c("/home/ayu/Desktop/projek-erlang/formula", [{outdir, "/home/ayu/Desktop/projek-erlang/"}]).
c("/home/ayu/Desktop/projek-erlang/formula", [{outdir, "/home/ayu/Desktop/projek-erlang/"}]).
{ok, formula}

10> formula:glbb({distance, 40, 5}).
{distance,200}
11> hello.
hello
12>
-U:*** *erlang* Bot L12 (Erlang Shell:run Compilation)-----

```

ketika kita ketikkan "hello" diakhiri dengan tanda titik, maka fungsi akan mengeluarkan kata yang sama "hello" karena dianggap atoms. Namun jika kita tulis hello dengan diawali huruf besar, maka kita sudah mencoba memanggil Variabel, dan variabel tidak bisa berdiri sendiri, melainkan harus ada nilai yang mengikatnya. Contoh:



```
File Edit Options Buffers Tools Errors Complete In/Out Signals Help
glbb({speed, Distance, Time}) ->
  {speed, Distance / Time};
glbb({distance, Speed, Time}) ->
  {distance, Speed * Time};
glbb({time, Speed, Distance}) ->
  {time, Distance / Speed}.
[]

--:-- formula.erl Bot L24 (Erlang)-----
c("/home/ayu/Desktop/projek-erlang/formula", [{outdir, "/home/ayu/Desktop/projek-erlang/"}]).
{ok, formula}

10> formula:glbb({distance, 40, 5}).
{distance,200}
11> hello.
hello
12> Hello = 15.
15
13> Hello.
15
14>
-U:*** *erlang* Bot L16 (Erlang Shell:run Compilation)-----
```

Hello = 15. ketika kita jalanin, maka Variabel Hello menjadi variabel terikat yang memiliki nilai 15. Begitu juga kalau kita hanya memanggil atoms Hello disertai dengan titik, fungsi akan langsung mengeluarkan nilai yang sudah tersimpan dalam sistem sebagai binded variable. Contoh: Hello. Maka yang keluar adalah nilai yang sudah mengikat variabel Hello, yaitu 15.

Demikian Langkah-langkah menggunakan Emacs dalam pembuatan modul sederhana dalam bahasa pemrograman Erlang.