Budi Rahardjo

Catatan Pemrograman BR

Catatan Pemrograman BR

ISBN–INFO
ISBN–13:

Pengantar

Seringkali saya mengerjakan pemrograman dan lupa akan sesuatu. Dulu saya pernah melakukan ini, tapi kok sekarang lupa lagi. Lantas saya mencari-cari di komputer saya, kalau-kalau ada contoh kode atau catatan yang pernah saya buat. Atau kemudian saya cari-cari di internet untuk contoh yang saya maksudkan.

Proses ini menghabiskan waktu. Daripada seperti itu, lebih baik saya catat di sini saja. Selain bermanfaat untuk saya, mudah-mudahan ini juga dapat bermanfaat bagi orang lain.

Tentu saja format dan isi yang saya tuliskan ini sesuai dengan kebutuhan saya. Misalnya, contoh perintah-perintah yang ditampilkan adalah apa-apa yang pernah atau biasa saya gunakan. Ada banyak perintah (command) dan variasinya yang tidak saya catat. Bagi saya, hal-hal yang khas (spesifik) saya ini yang justru perlu saya catat. Jadi ketika butuh, saya teringat pernah melakukan itu tetapi lupa *command line*-nya. Maka di sinilah fungsi dari buku atau catatan ini. Untuk yang belum pernah saya lakukan, ya saya akan cari dari internet atau membaca manualnya.

Catatan ini juga bukan pengganti manual atau buku yang spesifik terhadap topik tersebut. Untuk bagian bahasa Perl, misalnya, buku ini bukan pengganti buku *Camel* (buku dari O'Reilly yang sampulnya bergambar onta). Buku itu lebih komprehensif dari catatan ini. Namun fungsinya memang beda. Yang ini adalah sebagai catatan dari hal-hal yang sering saya lakukan (lupa).

Topik yang masuk ke kategori "programming" ini cukup luas. Jadi ada banyak hal yang mungkin agak sedikit melebar, meskipun bisa juga dikait-kaitkan. Sebagai contoh ada pembahasan tentang `git` (*source code versioning*) dan database (`Postgres`). Bahkan pada versi awal dari catatan ini, yang betulan membahas tentang koding malah lebih sedikit.

Saya suka menggunakan *command line*. (Bahkan untuk kodingpun saya masih suka menggunakan editor `vi`) Untuk itu contoh-contoh yang saya tampilkan adalah yang menggunakan *command line*. Alternatif cara yang menggunakan GUI (graphical user interface) sebetulnya ada dan banyak. Mungkin nanti saya berikan tautan ke halaman atau informasi yang berhubungan dengan itu.

Platform saya juga bervariasi. Umumnya saya menggunakan Linux mint (untuk desktop), Mac OS X (untuk portable / jalan), dan Linux Debian (dalam virtualbox di Mac OS) Ada sedikit kerepotan tentang tools yang tersedia di platform yang berbeda-beda. Untuk itu, saya berusaha menggunakan tools yang multi-platform. Biasanya tools yang multi-platform bersifat *command line*. Tidak terlalu masalah bagi saya.

Selamat menikmati versi 0.2 ini.

Bandung, 2016
Budi Rahardjo
@rahard
Budi Rahardjo, *"Catatan Pemrograman BR"*, PT. Insan Infonesia, 2016.

Bab 1

Perl

Perl adalah bahasa pemrograman kesukaan saya. Pada mulanya, sekitar akhir tahun 80-an, Perl merupakan satu-satunya bahasa pemrograman yang *portable*. Saya menggunakan banyak komputer dengan sistem operasi yang berbeda-beda; MS-DOS, UNIX dan variasinya (SunOS, Solaris, AIX, SCO UNIX, HP-UX, Linux, FreeBSD, OpenBSD). Sesungguhnya ada bahasa lain yang juga tersedia di berbagai platform , yaitu bahasa *C*, tetapi bahasa C harus dirakit (*compile*) terlebih dahulu, sementara bahasa *Perl* adalah intrepreter yang langsung dapat dieksekusi tanpa melalui proses rakit. Siklus pengembangan program menjadi lebih cepat.

Cara saya memprogram dalam bahasa Perl, sesungguhnya seperti cara saya membuat program program dalam bahasa *C*. Saya lebih menekankan aspek *readability* daripada membuat program yang singkat dan padat. Padahal banyak programmer Perl yang cenderung membuat programnya singkat, bahkan sebaris yang disebut *one-liner*.

Saya sering menggunakan bahasa Perl. Banyak hal yang sudah teringat di kepala saya sehingga tidak membutuhkan catatan ini. Oleh sebab itu mungkin saja catatan ini terlalu "meloncat".

Salah satu yang paling dibenci oleh orang (bukan programmer Perl) terhadap bahasa Perl adalah penggunaan dolar ($) atau karakter lainnya (@, !, dan seterusnya) untuk variabel yang membuat kode terlihat seperti banyak cacing. ha ha ha.

1.1 Variabel

Variabel di perl ada beberapa jenis dan ini ditunjukkan dengan karakter di depan nama variabelnya.

- Dolar `$` untuk skalar. Contoh `$x = 'A'`, `$x[0]=123`, dan seterusnya. Tipe dari variabel tersebut bebas, bisa numerik (integer) atau string. (Ini enaknya perl dan juga sekaligus bahayanya.)
- Tanda *at* `@` untuk array. Contoh `@x`. Elemen dari array tersebut dapat diakses dengan tamda dolar `$x[i]`.
- Persen (%) untuk *associative array*. Contoh `%x`. Elemen diakses dengan menggunakan `$x{key}`. Ini adalah salah satu hal yang paling asyik dari perl. Saya bisa

menggunakan apa saja (umumnya string) sebagai *key* dari *associative array* ini. Contoh `$header{Nama}="Budi Rahardjo"`, `$header{Twitter}="@rahard"`

1.2 Loop

Salah satu hal yang sering dilakukan oleh sebuah program adalah membuat pengulangan (loop). Jika kita mengetahui jumlah loop, maka biasanya *for-loop* merupakan construct yang paling lazim. Bagi yang familier dengan bahasa *C*, bentuk ini mirip sekali dengan loop di bahasa *C*.

```
for ($i=0; $i<10; $i++) {
   print "$i ... ";
}
```

Look juga digunakan untuk memproses sebuah berkas, membaca baris per-baris dari baris pertama hingga akhir berkas (EOF). Jika kita memiliki skrip `baca.pl` dengan isi seperti di bawah ini.

```
while (<>) {
   print $_;
}
```

Kemudian dijalakan dengan memberikan nama berkas yang akan diproses, `berkas-data.txt`. Berkas tersebut akan dibaca baris per-baris dan akan ditampilkan (print). Variabel `$_` berisi baris yang dibaca.

```
perl baca.pl berkas-data.txt
```

1.3 Topic Generator

Berikut ini adalah sebuah contoh skrip yang saya beri nama `generator.pl`. Skrip ini terinspirasi dari kesulitan orang dalam mencari topik untuk menulis di blognya. Skrip ini menampilkan (generate) topik. Sesungguhnya skrip ini hanya membaca topik-topik yang sudah dituliskan dalam berkas `topics.txt`, kemudian memilih secara random salah satu dari baris (topik) tersebut. Setiap topik dipisahkan dengan dua garis (dash).

Contoh (cuplikan) isi berkas `topics.txt` adalah sebagai berikut. Anda tinggal menambahkan topik baru ke dalam berkas tersebut. Skrip akan langsung menggunakannya sebagai bagian dari pilihan.

```
Ceritakan tentang buku yang paling berkesan kepada Anda
--
Ceritakan tentang media sosial yang paling sering Anda gunakan.
Mengapa Anda sering menggunakan itu?
--
Jika Anda menjadi superhero, siapa yang Anda pilih?
--
```

Berikut ini adalah kode `generate.pl` tersebut. Penjelasan akan saya berikan di bawah.

```
1  #! /usr/bin/perl
2  $topicDB='topics.txt';
3  open(my $DB, '<', $topicDB) or die $!;
4  $count=0;
5  while (<$DB>) { if ($_ =~ "^--") { $count++; }
6  else { $topic{$count} = $topic{$count} . $_;} }
7  close($DB);
8
9  $luckyone =  int(rand($count-1));
10 print $topic{$luckyone};
11 exit;
```

Baris pertama hanya menunjukkan dimana interpreter perl berada. Baris 2, set nama berkas topik ke variabel `$topicDB`. Baris 3, membuka berkas tersebut dalam mode read (lihat tanda $<$) dan apabila gagal akan menampilkan pesan (`die $!`). Ini adalah salah satu hal yang menarik bagi saya, yaitu error message akan ada pada variabel itu. Tinggal di-print saja. (Tentu saja ini bagus untuk command line script, tapi tidak bagus kalau dijadikan aplikasi web-based.)

Baris ke 4, menyiapkan diri untuk memulai menghitung jumlah topik (ke dalam variabel `$count`). Baris ke 5, melakukan looping terhadap berkas topik itu. Jika menemukan baris yang dimulai dengan garis dua (–) maka counter perlu dinaikkan. Baris ke 6, memasukkan kalimat (baris) ke dalam associative array `%topic` (dengan nomor topik sebagai key). Dia dibuat begitu agar kalau ada topik yang lebih dari satu baris, digabung menjadi satu variabel.

Baris ke 9, memilih salah satu dari topik secara random. Angka random dipilih maksimal sama dengan jumlah topik. Baris ke 10, menampilkan di layar. Mudah bukan?

1.4 To Do

Hal-hal yang belum saya catat adalah tentang *search and replace* dan *regular expression* (regex).

Bab 2

Postgres

Postgres (postgresql) merupakan program database yang banyak digunakan untuk aplikasi internet. Ini merupakan saingan dari MySQL.

Untuk mengakses database Postgres dari shell dapat digunakan program *psql*. Ini merupakan bagian dari klien Postgress[1].

```
psql -U username -h 192.168.1.1 -d databasename
```

Untuk menjalankan kode (perintah SQL) yang ada dalam berkas "perintah.sql" dapat dilakukan dengan cara di bawah ini. Keluaran akan ditampilkan di *stdout*, yang kemudian bisa di-pipe ke program lain (atau diarahkan ke file).

```
psql -U username -h 192.168.1.1 -d databasename -f perintah.sql
psql -U username -h 192.168.1.1 -d databasename -f perintah.sql | more
psql -U username -h 192.168.1.1 -d databasename -f perintah.sql > out.
    txt
```

Isi berkas "perintah.sql" adalah seperti ini:

```
select * from auth_user
```

[1] (Untuk klien di Mac OS X, saya masih belum menemukan yang pas karena saya menggunakan *Homebrew* sebagai package manager. Ternyata klien Postgres adanya di Fink. Sementara ini saya menggunakan *PSequel*, yang GUI, untuk Mac OS X.)

Bab 3

Git

Git[1] adalah sebuah sistem untuk melakukan *versioning* dari dokumen, yang biasanya adalah kode sumber (*source code*). Biasanya *git* digunakan untuk mengembangkan software secara bersama-sama, tetapi dia dapat digunakan untuk keperluan lain. Penulisan buku ini menggunakan *git* untuk sinkronisasi antar komputer tempat saya bekerja.

Selain untuk mengembangkan kode bersama, git juga bermanfaat untuk mengembangkan kode di komputer yang berbeda. Di komputer A, saya bisa bekerja. Setelah selesai, dokumen saya simpan (*push*) ke server git. Di komputer B, saya bisa mengambil berkas atau perubahan terbaru dengan melakukan *pull*. Maka pekerjaan di kedua komputer tersebut menjadi tersinkronisasi dengan *git* sebagai penengah.

Mari kita mulai menggunakan *git*. Buat sebuah direktori tempat bekerja. Pindah ke direktori tersebut dan melakukan inisialisasi *git*. Pada direktori tersebut akan dibuat direktori `.git` yang berisi informasi mengenai pekerjaan kita. Untuk saat ini, kita tidak perlu tahu detail dari isi direktori itu.

```
git init
ls -a
```

Untuk mengetahui konfigurasi git Anda secara global dapat digunakan perintah *list* berikut. Untuk mengubah konfigurasinya juga dapat dilakukan dengan perintah `git config`. Konfigurasi ini hanya perlu dilakukan sekali saja.

```
git config --list
git config --global user.name "budi rahardjo"
git config user.name
```

Selanjutnya kita mulai menambahkan server git (dalam contoh ini IP-nya adalah `192.168.1.1` - untuk kasus Anda coba cari alamat ini, misal di github.com ada di sebelah kanan atas). Kemudian kita dapat menarik kode dengan perintah *pull*.

```
git remote add origin http://192.168.1.1/nama-proyek/proyeknya.git
git pull origin master
```

[1] Client git dapat diperoleh dari https://git-scm.com/
Sementara server git dapat menggunakan Gogs yang dapat diperoleh di https://gogs.io/

Boleh jadi kita memulai dengan melakukan *clone* juga. Contoh di bawah ini kita akan membuat direktori `libgit2` dan mengambil semua berkas yang ada di sana.

```
git clone https://github.com/libgit2/libgit2
```

Sekarang kita boleh bekerja dengan berkas-berkas. Setelah selesai, berkas perubahaan kita bisa ditambahkan dengan memberikan perintah *add*.

```
git add somefile
git add *.c
git add LICENSE
git commit -m 'initial project version'
```

Pilihan `-m` tersebut akan mendokumentasikan pesan ke dalam perubahan. Jika kita tidak menggunakan opsi tersebut, kita akan dibawa ke editor pilihan kita.

Setor perubahan ke server git.

```
git push origin master
```

Untuk memulai lagi, misal hari berikutnya untuk bekerja. Contoh berikut menunjukkan bahwa apa yang ada di direktori kita sama dengan ada yang di (remote) server.

```
git pull origin master
From https://github.com/rahard/buku-catatan-br
 * branch            master     -> FETCH_HEAD
Already up-to-date.
```

Sekali-sekali saya cek apakah saya membuat perubahan yang perlu saya setor. Ada daftar berkas yang berubah dan perlu saya tambahkan dengan perintah add sebelum di-commit.

```
git status
git add fileyangberubah
git commit -m 'perubahan ini tentang apa'
```