Pengenalan Pemrograman Ruby

Fajar Muharandy
muharandy@yahoo.com

Lisensi Dokumen:
Copyright © 2003-2007 IlmuKomputer.Com
Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.

Tutorial ini merupakan pengenalan awal bagi anda yang belum pernah melakukan pemrograman dengan Ruby. Namun, hal-hal yang diajarkan di dalam tutorial ini tidak akan membahas konsep-konsep dalam bahasa Ruby secara detail. Malah, bisa saya bilang bahwa tutorial ini lebih tepat dijadikan "pendamping" bagi anda yang ingin membaca tutorial *Pengenalan Ruby on Rails*[1], ketimbang menjadikan tutorial ini sebagai referensi untuk belajar Ruby secara menyeluruh.

Apa yang saya tulis di sini kebanyakan merupakan hal-hal dimana saya *tersandung* ketika mempelajari Ruby di awal. Sehingga, saya harap pengalaman saya ini dapat dijadikan pelajaran berharga bagi anda semua.

Penulis

[1] Saya juga menulis tutorial Pengenalan Ruby on Rails, yang bisa anda dapatkan di website tempat anda mendownload tutorial ini juga.

Daftar Isi

Daftar Isi 2
1. Variable 3
2. Array dan Hash 4
3. Control Structure 6
4. Method 7
5. Classes 8
6. Kemana Setelah Ini 10
Lampiran A: Menginstal Ruby Pada Windows 11
Lampiran B: Biografi Penulis 13

Copyright © 2003-2007 IlmuKomputer.Com

1. Variable

Berbeda dengan bahasa pemrograman seperti C dan Java, di dalam Ruby kita dapat langsung mendefinisikan sebuah variable tanpa menentukan tipenya. Anda dapat mencobanya langsung melalui *irb*[2] .

```
$irb> a = 2
$irb> b = 2
$irb> a + b
$=> 4
```

Bukan hanya tipe data numeric, anda juga bisa membuat tipe data string secara langsung.

```
$irb> a = "hehe"
$irb> a.length
$=> 4

$irb> a.reverse
$=> "eheh"

$irb> a
$=> "hehe"
```

Di dalam Ruby kita mengenal istilah *symbol*. *Symbol* ini akan sering sekali anda temui di dalam Rails. Sering digunakan sebagai semacam konstanta pengganti string.

```
$irb> a = :test
$irb> a != :test
$=> false

$irb> a == :test
$=> true
```

Symbol ini bisa dibilang lebih hemat memori dibandingkan dengan String. Di dalam Ruby kita akan sering menemukan *symbol* dalam sebuah pemanggilan method. Selain itu kita juga akan sering menggunakan *symbol* sebagai sebuah *key* dalam *hash*. Kedua hal ini sering membuat bingung orang-orang yang pertama kali mempelajari Ruby.

[2] Interactive Ruby. Bisa digunakan untuk mempelajari Ruby tanpa harus membuat file .rb terlebih dahulu. Anda menjalankannya dengan mengeksekusi perintah *irb* pada console

Copyright © 2003-2007 IlmuKomputer.Com

2. Array dan Hash

Bagi anda yang sudah akrab dengan pemrograman pasti sudah mengenal apa yang dimaksud dengan *array* dan *hash*. Di dalam Ruby saya bisa mengatakan bahwa *array* sangat terkait dengan simbol [] sedangkan *hash* dengan {}. Mengapa? Karena memang itulah salah satu cara untuk menginisialisasi *array* dan *hash*.

```
$irb> a = []
$=> []
$irb> a = Array.new
$=> []

$irb> a = [47, 77, 17]
$=> [47, 77, 17]

$irb> a << 107
$=> [47, 77, 17, 107]

$irb> a.length
$=> 4

$irb> a.sort
$=> [17, 47, 77, 107]
```

Pada contoh diatas terlihat bahwa kita bisa juga melakukan inisialisasi *array* dengan memanggil method *new()*. Selanjutnya kita bisa menambahkan elemen pada *array* dengan menggunakan operator <<.

Berbeda dengan *array*, pada *hash* kita memiliki pasangan *key* dan *value*. Sebagaimana yang telah saya sebutkan sebelumnya, dalam Ruby *hash* ini seringkali dipakai sebagai parameter dalam pemanggilan sebuah method.

```
$irb> a = {}
$=> {}
$irb> a = Hash.new
$=> {}

$irb> a = {:keren => “abis”}
$=> {:keren => “abis”}
$irb> a[:manthab] = “jaya”
$=> “jaya”

$irb> a[:keren]
$=> “abis”
$irb> a[:manthab]
$=> “jaya”
```


Copyright © 2003-2007 IlmuKomputer.Com

Pada contoh di halaman sebelumnya terlihat bahwa kita bisa menggunakan *symbol* sebagai *key*. Namun, ternyata kita juga bisa melakukan hal yang sebaliknya sebaimana yang terlihat pada contoh berikut ini.

```
$irb> a[“keren”] = :jaya
$=> :jaya

$irb> a
$=> {:keren=>"abis", "keren"=>:jaya, :manthab=>"jaya"}
```

Sama seperti *array* kita bisa menggunakan method *length()* untuk melihat jumlah data yang ada di dalam *hash*.

```
$irb> a.length
$=> 3
```


Copyright © 2003-2007 IlmuKomputer.Com

3. Control Structure

Sekarang kita akan mempelajari *control structure* di Ruby. Sampai sini kita sebenarnya masih bisa menggunakan *irb.* Namun agar lebih jelas dan mudah dimengerti kita akan membuat sebuah file Ruby bernama *coba1.rb*.

```
a = ARGV[0].to_i

if a > 7
  puts "#{a} lebih besar dari 7"
elsif a < 7
  puts "#{a} lebih kecil dari 7"
else
  puts "#{a} sama dengan 7"
end
```

Perhatikan bahwa Ruby mengenali *elsif* bukan *else if.* Pada kode diatas kita melakukan substitusi nilai *a* di dalam tanda kutip dengan cara *"#{a}"*. Sekarang coba jalankan program di atas dan masukkan nilai sembarang sebagai argumen.

```
$ruby coba1.rb 8
$8 lebih besar dari 7
```

Kita akan sering menggunakan *looping* di dalam aplikasi kita. Salah satu caranya adalah dengan menggunakan *for*. Sekarang mari kita buat file baru *coba2.rb* yang di dalamnya kita isi dengan *code* berikut:

```
for i in 1..10 do
  puts "#{i}"
end
```

Ketika dijalankan, program ini akan mencetak angka 1 sampain dengan 10 ke layar.

Selain *for* kita juga bisa menggunakan while untuk melakukan *looping*. Berikut ini adalah contoh penggunaannya dalam file *coba3.rb*

```
i = 1
while i <= 10
  puts i
  i = i + 1
end
```

Ketika dieksekusi program ini akan memberikan output yang sama dengan *coba2.rb*.

Copyright © 2003-2007 IlmuKomputer.Com

4. Method

Sebuah *method* dideklarasikan dengan menggunakan *keyword* *def* dan diakhiri dengan *end*. Sebagai contoh buatlah file baru *coba4.rb.*

```
def coba_method(name)
   puts "coba #{name}"
end

a = "Jaya!"
coba_method(a)
coba_method("Dunks!")
```

Ketika dieksekusi, program diatas akan mengahasilkan output sebagai berikut:

```
coba Jaya!
coba Dunks!
```

Untuk me-*return* sebuah nilai, kita hanya perlu menuliskan nilai (atau variable) tersebut di dalam method.

```
def kali_dua(nilai_awal)
  kali_dua = nilai_awal * 2
  kali_dua
end

kali_dua = kali_dua(4)
puts "kali dua #{kali_dua}"
```

Hasil eksekusi dari program ini adalah:

```
kali dua 8
```

Sebenarnya bisa saja kita menggunakan *keyword return* untuk mereturn sebuah nilai dari method. Namun, di sini saya hanya ingin menunjukkan alternatif dari cara tersebut.

Copyright © 2003-2007 IlmuKomputer.Com

5. Classes

Deklarasi sebuah kelas dimulai dengan *class* dan diakhiri dengan *end.* Sebagai contoh kita akan membuat sebuah kelas Mahasiswa pada file mahasiswa.rb.

```
class Mahasiswa
  def initialize(nama, npm)
    @nama = nama
    @npm = npm
  end
  def cetak_info
    puts "Nama: #{@nama}"
    puts "NPM: #{@npm}"
  end
end
```

Instance variable dari sebuah kelas dideklarasikan dengan '@' yang akan membuatnya bisa diakses oleh semua *instance method* yang ada di kelas tersebut. Namun *instance variable* tersebut tidak akan bisa diakses dari luar kelas. Untuk bisa melakukan itu kita perlu membuat *accessor* method. Tambahkan kedua *method* ini pada kelas Mahasiswa yang telah kita buat.

```
def nama=(nama_baru)
  @nama = nama_baru
end
def nama
  @nama
end
```

Anda bisa menguji method ini dengan melakukan pemanggilan method seperti dibawah ini.

```
mhs = Mahasiswa.new("Si Bocung","1203000439")
mhs.cetak_info

mhs.nama = "Si Bocah" # mengubah nama menjadi Si Bocah
mhs.cetak_info

nama = mhs.nama # method ini akan mereturn Si Bocah sebagai nama
puts nama
```

Namun untuk mendefinisikan method *setter* dan *getter* bagi setiap variable adalah pekerjaan yang melelahkan. Oleh karena itu Ruby menyediakan method *attr_accessor()* untuk memudahkan kita.

```
class Mahasiswa
  attr_accessor :npm
  attr_accessor :nama
```


Copyright © 2003-2007 IlmuKomputer.Com

Selain *accessor* method ada lagi yang akan sering anda temukan di Rails, yaitu *class* method.

```
def self.whatami
  puts “Saya Mahasiswa”
end
```

Method ini diakses dengan diawali oleh nama kelasnya.

```
Mahasiswa.whatami    # akan mencetak Saya Mahasiswa
```

Ada banyak hal lain mengenai *class* namun tidak bisa di bahas dalam tutorial ini seluruhnya. Anda bisa mempelajari hal-hal tersebut melalui link-link yang saya sediakan pada halaman selanjutnya**.**

Copyright © 2003-2007 IlmuKomputer.Com

6. Kemana Setelah Ini

Pada akhir tutorial ini anda telah mempelajari beberapa konsep di dalam bahasa Ruby. Saya tekankan sekali lagi bahwa tutorial ini hanyalah pengenalan Ruby sebelum anda mempelajari Rails. Namun, saya tidak akan melarang anda untuk mempelajari Ruby lebih lanjut setelah ini. Berikut ini adalah link website-website yang berguna untuk mempelajari Ruby.

1. ***http://poignantguide.net/ruby/***
   - Belajar Ruby lewat novel dan komik.
2. **http://tryruby.hobix.com**
   - Tutorial Ruby yang sangat bagus. Anda bisa langsung mencobanya pada browser anda.
3. **http://www.ruby-lang.org/**
   - *Ruby: Programmers' Best Friend*. Ini adalah salah satu slogan dari website ini. Di sini anda bisa banyak hal mengenai Ruby termasuk link-link ke website Ruby yang lain.

Copyright © 2003-2007 IlmuKomputer.Com

Lampiran A: Menginstal Ruby Pada Windows

Tutorial ini akan menjelaskan langkah-langkah untuk menginstal Rails di komputer berbasis windows. Contoh yang akan diberikan dalam tutorial ini adalah cara menginstal Ruby on rails pada Windows 2000.

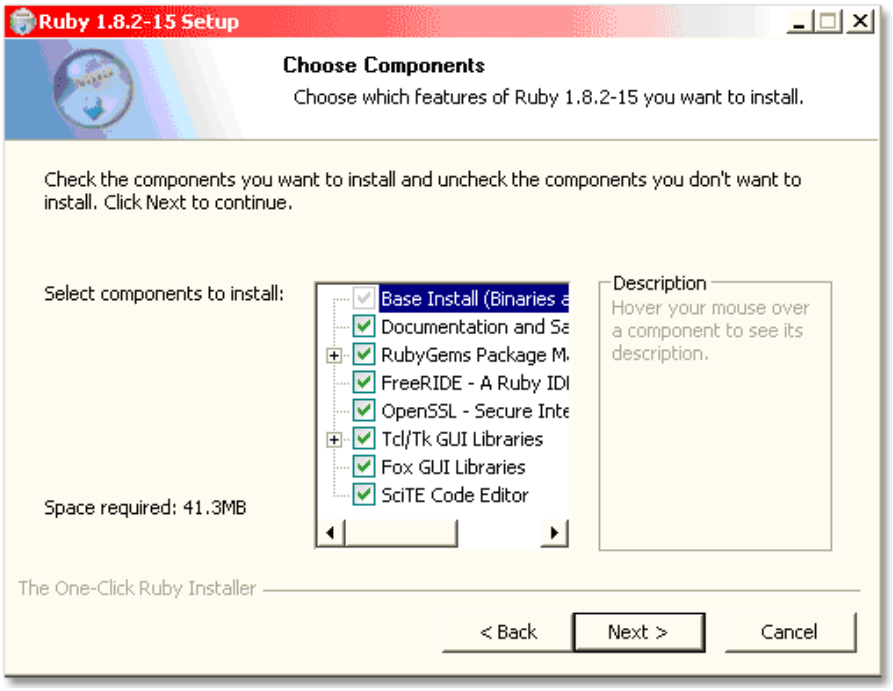
1. Download Ruby Installer for Windows

Anda bisa mendownload installer Ruby[3] ini di http://www.rubyforge.org[4]

Versi terbaru dari ruby installer pada saat tutorial ini ditulis adalah Ruby1.8.2-15. Anda sangat disarankan untuk menggunkan versi terbaru yang stabil.

2. Jalankan Installer Ruby

Ikuti proses installasi yang ada, anda mungkin akan memilih untuk tidak menginstall beberapa komponen yang ada jika memang anda tidak memerlukan. Tapi jika anda tidak tahu-menahu apa yang anda lakukan, dan anda tipe orang yang tidak memiliki masalah dengan *free space* di HardDisk anda, anda bisa menginstal semuanya.



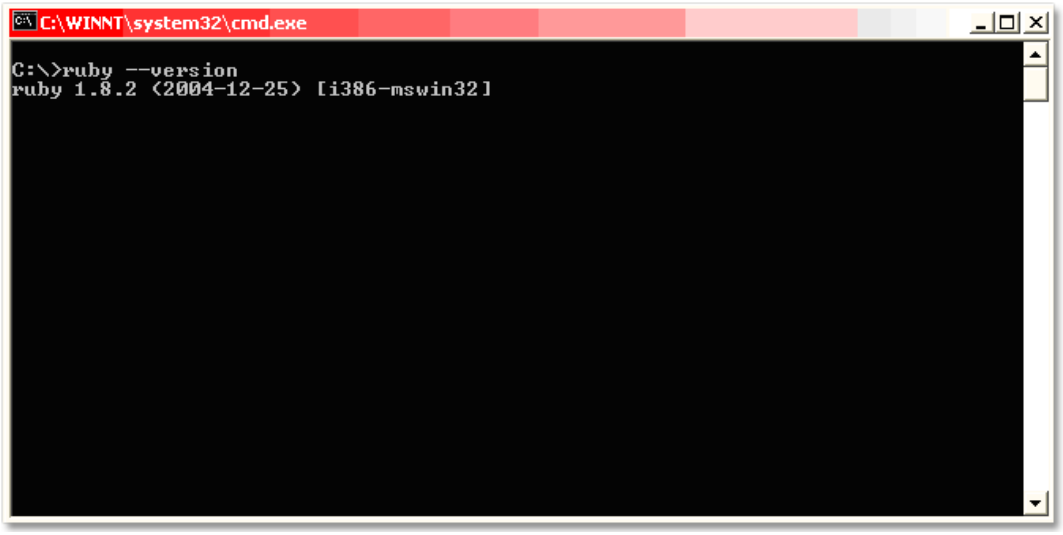
Gambar 1 - Proses Instalasi Ruby

[3] Halaman project ruby installer untuk windows adalah http://rubyforge.org/projects/rubyinstaller/
[4] http://www.rubyforge.org merupakan tempat diamana anda dapat mendownload distribusi Ruby dan juga extensionnya misalnya seperti driver untuk database PostgreSQL yang tidak diikut sertakan dalam installer Ruby.

Copyright © 2003-2007 IlmuKomputer.Com

3. Memastikan Ruby Telah Terinstal Dengan Sukses

Untuk memastikan bahwa Ruby telah terinstall dengan sukses dapat dilakukan dengan cara menjalankan perintah ruby --version pada konsol.



Gambar 2 - Melihat Versi Ruby

Jika anda berhasil, berarti anda telah sukses menginstall Ruby di Komputer anda.

Copyright © 2003-2007 IlmuKomputer.Com

Lampiran B: Biografi Penulis

Fajar Muharandy, mahasiswa Fakultas Ilmu Komputer UI angkatan 2003. Pernah menjabat sebagai ketua SIG Information System RiSTEK CSUI periode 2005-2006. Pada periode yang sama juga pernah menjadi kontributor *project runes* Enterprise Application Lab CSUI. Beberapa artikel dan tutorial yang ditulis di sini merupakan hasil kerjanya dalam *project runes.* Bidang minatnya adalah desain grafis, komputasi grid, dan E-Government. Sangat senang membaca dan menulis ^__^.

Feel free to contact me at:

e-mail : muharandy[at]yahoo[dot]com

Y!M : muharandy

Copyright © 2003-2007 IlmuKomputer.Com