```ruby
#!/usr/bin/env ruby

class Ruby
  attr_accessor :programming_language

  def to_s
    @programming_language
 end
end

 rb = Ruby.new
 rb.programming_language = "ruby"

 puts rb.to_s
```

puts "  "

LINESIP

מה היא רובי ?

- רובי היא שפת תכנות דינאמית, המכילה השפעות מהשפות Perl ו SmallTalk
- השפה נוצרה על ידי יוקיהירו מאטסוטו (松本行弘) באמצע שנות ה90 של המאה הקודמת
- במקור השפה קיבלה את השם MRI – Matz's Ruby Interpreter
- השפה תומכת בשתי פרדיגמות של תכנות:
  - תכנות פרוצדורלי (procedural programming)
  - תכנות מונחה עצמים (object oriented programming)

משתנים גלובליים:

דוגמאות למשתנים גלובליים שהוגדרו מראש (משתנה גלובלי מתחיל תמיד עם הסימן $):

$! - מידע על החריגה שהתקבלה

$@ - מידע איתור לאחור של חריגה

$DEBUG - מצב הניפוי שגיאות שהועבר באמצעות המתג d-

$FILENAME - שם הקובץ הנוכחי

$LOAD_PATH - נתיב טעינת המודולים השונים

$VERBOSE - מאפשר לדעת אם להציג עוד מידע במידע ו v- הועבר

$stderr - מכיל את ההפנייה לפלט השגיאות הנוכחי

$stdout - מכיל את ההפנייה לפלט הנוכחי

$stdin - מכיל את ההפנייה לקלט הנוכחי

כיצד נראה התחביר?

תכנות מונחה עצמים:

```
class AClass
   def initialize(value)          # constructor
     @instance_variable = value   # instance variable
     @@class_variable = value     # class variable
   end

   def instance_variable=(value) # Set
     @instance_variable = value
   end

   def instance_variable # Get
     @instance_variable
   end
end
```

כיצד נראה התחביר?

תכנות מונחה עצמים:

```
class AClass
   attr_accessor :instance_variable # or attr_reader and attr_writer

   def initialize(value)          # constructor
     @instance_variable = value   # instance variable
     @@class_ariable = value      # class variable
   end

   def instance_variable=(value) # Set
     @instance_variable = value
   end

   def instance_variable # Get
     @instance_variable
   end
end
```

כיצד נראה התחביר?

הכירות קצרה עם Iterators:

```
array = [1, 2, 3, 4]
```

ב PHP נרוץ עליו בצורה הזו:

```
foreach ($array as $item)
{
   print $item . "\n";
}
```

ברובי נרוץ עליו בצורה הזו:

```
array.each do |item|
  puts item
end
```

הרחבת מחלקה

ככה מרחיבים מחלקה ברובי:

```
class String
  def rot13(s)
    hash = {“a” => “n”, “b” => “o”, “c” => “p” ...}
    result = “”
    s.each_byte do |ch|
      result += hash[sprintf(“%c” ,ch)]
    end
    result
  end
end
```

נשתמש בהרחבה בצורה הבאה:

```
puts String.new.rot13(“Hello world”)
= Uryyb jbeyq
```

הרחבת מחלקה

יש דרך טובה יותר:

```
class String
  def String.rot13(s)
    hash = {“a” => “n”, “b” => “o”, “c” => “p” ...}
    result = “”
    s.each_byte { |ch| result += hash[sprintf(“%c” ,ch)] }
    result
  end
end
```

השימוש עכשיו יהיה:

```
puts String.rot13(“Hello world”)
= Uryyb jbeyq
```

מודול ברובי:

השימוש של מודולים (module) ברובי קיים על מנת:

- להרחיב מחלקה
- לתת אפשרות לבצע "ירושה מרובה", בלי באמת לקבל תמיכה כזו

המאפיינים של מודול ברובי:

- חוסר יכולת לרשת ממחלקות/מודולים אחרים בניגוד למחלקה
- לא קיים מופע (instance) למודולים

מודול ברובי:

ככה משתמשים במודול:

```
module Debug
    include Log  # include another module named log
    def initialize(name)
      @name = name
    end

    attr_reader @name
end

class Logger
    include Debug    # First module to include
    include Math     # Another module to include

    def add(s)
     ...
    end
end

logger = Logger.new("main")
logger.add("Adding to logger")
puts logger.name
```

מודול ברובי:

ככה משתמשים במודול:

```
# Constants
puts Math::PI
= 3.14159265358979

# Method
puts Math.sqrt(100)
= 10.0
```

שימושים מעשיים ברובי:

יצירת מחלקה שהיא Singleton:

```
require "singleton"
class << String
  include "Singleton"
  ...
end
```

בואו נחזור למתודה הבאה:

```
def String.rot13(s)
  hash = {"a" => "n", "b" => "o", "c" => "p" ...}
  result = ""

  s.each_byte { |ch| result += hash[sprintf("%c" ,ch)] }

  result
end
```

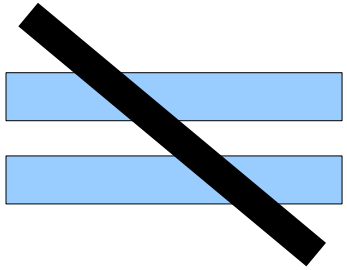
כך ניתן לכתוב את אותו הדבר:

```
class << String
  def rot13(s)
    ...
  end
end
```

שימושים מעשיים ברובי:

עוד על Singleton:

```
class Foo
   @a = 1
end
```



```
class Bar
  def initialize
    @a = 1
  end
end
```

ההבדל:

```
Foo.new.instance_variables
= []

Bar.new.instance_variables
= ["@a"]
```

שימושים מעשיים ברובי:

שדונים ברובי:

my_daemon.rb

```
#!/usr/bin/ruby
import "rubygems"
import "daemons"

Daemons.run("code.rb")
```

code.rb

```
#!/usr/bin/ruby
loop do
  puts "Hello world"
  sleep(100)
end # loop do
```

תוצאה:

```
Usage: code.rb <command> <options> -- <application options>

* where <command> is one of:
  start          start an instance of the application
  stop           stop all instances of the application
  restart        stop all instances and restart them afterwards
  run            start the application and stay on top
  zap            set the application to a stopped state

* and where <options> may contain several of the following:

    -t, --ontop                      Stay on top (does not daemonize)
    -f, --force                      Force operation

Common options:
    -h, --help                       Show this message
        --version                    Show version
```

מה עוד לדעת:

אף שעדיין לא התחלנו לגעת ברובי, כבר עכשיו אפשר להבין את הכוח

והעוצמה שהשפה מציעה לנו.

הנה עוד מספר דברים שדי טוב לדעת ו/או להיזכר בהם:

Name convention:

- ClassNames
- method_names and variable_names
- methods_asking_a_question?
- method_setter=
- slightly_dangerous_methods!
- @instance_variables
- SOME_CONSTANTS or OtherConstants

מה עוד לדעת:

למה אין צורך בממשק (interface) ברובי ?

```
class Dog
  def sound
    "wof"
  end
end

class Wolf
  def sound
    "Ah whow"
  end
end

k9 = [ Dog.new, Wolf.new ]

k9.each do |dog|
  puts dog.sound
end
```

כלי עזר בתכנות בRuby

ישנם מספר כלי עזר ברובי שיכולים לעשות לנו את החיים הרבה יותר קלים ופשוטים:

- rake - כלי המזכיר את Gnu Makefile
- ri - עזרה הדומה מאוד ל man או perldoc
- irb - שורת פקודה (shell) המאפשרת לנו להריץ פקודות ruby
- gem - התקנת מודולים עבור ruby

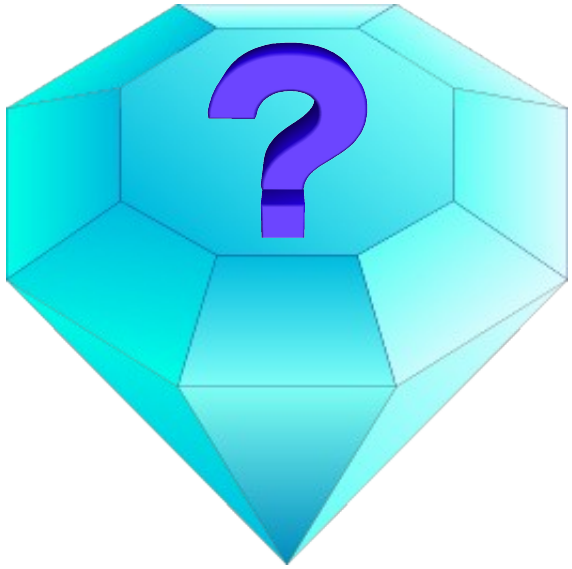
מה עוד לבדוק ?

ישנם כמה אתרים שמאוד מומלץ לבקר ולבדוק אותם:

- http://ruby-lang.org/
- http://ruby-doc.org/
- http://rubyforge.org/
- http://whytheluckystiff.net/ruby/pickaxe/
- http://fhwang.net/2004/11/14/Coming-to-Ruby-from-Java
- http://onestepback.org/articles/10things/index.html

- IRC – Freenode #ruby-lang

שאלות ?



רישיון המצגת