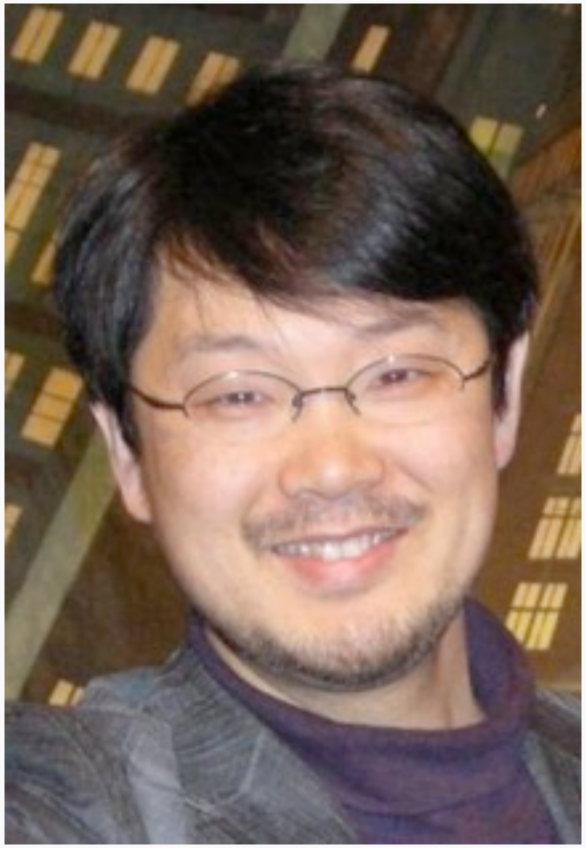
Ruby 101

Thomas Kjeldahl Nilsson
thomas@kjeldahlnilsson.net
linkedin.com/in/thomaskjeldahlnilsson
twitter.com/thomanil



“...we need to focus on humans, on how humans care about doing programming.”

-Yukihiro Matsumoto

What Are We Covering Today?

bare basics
trying it out
scripting, not system development
getting started!

Practical Info

Let’s Start!

Install

http://www.ruby-lang.org/en/downloads/

Run Code

```
ruby -e 'puts 123'
ruby myscript.rb
```

or from your editor

Explore

irb

ri

Basic Syntax

Difference From Java, C#

No need for compilation

No casting, no type declarations

Dude.new("Lebowski") instead of *new Dude("Lebowski")*

nil instead of *null*

variable_name instead of *variableName*

can often skip semicolons and parens

Methods

```
def greet(name)
  puts("Hello, "+name);
end

def greet name
  puts "Hello, "+name
end

greet(“Mr Smith”);       => “Hello, Mr Smith”

greet “Mr Smith”          => “Hello, Mr Smith”
```

Variables

```
question = “Meaning of life?”

the_answer = 42
answer_alias = the_answer
answer_alias        => 42
```

Objects

“Some string”.methods => *["upcase!", "zip", "find_index" ...]*

42.methods => *["%", "odd?", ...]*

Everything is an object.

Objects are easy to inspect.

Flow Control

```
if true
    puts “This is always true”
elsif false
    puts “This is always false”
else
    puts “This will never come to pass”
end
```

Flow Control

```
puts “Another way” if true

puts “This is never printed” unless false
```

EXERCISE 1

courseware/exercises/1

Iterators and Blocks

```
5.times do
    puts “Nice for-loop, eh?”
end

5.times { puts “One-liner block form” }

(1..3).each do |n|
    puts “Say hello to number #{n}!”
end

[1,2,3].map do |n|
    n * 2
end                          => [2, 4, 6]
```

EXERCISE 2

courseware/exercises/2

Range

```
(1..5).to_a   => [1, 2, 3, 4, 5]
(1...5).to_a  => [1, 2, 3, 4]
numbers = 0..9
numbers.max                  => 9
numbers.min                  => 0
numbers.include? 5          => true
```

Array

```
an_array = [1, "two", 3, “four”]   can have any types
an_array[0]        => 1
an_array[1..2]   => [“two”, 3]
an_array.first      => 1
an_array.last      => “four”

“A short sentence”[2..6]             => “short”
```

EXERCISE 3

courseware/exercises/3

String

```
holiday = “Christmas”
greeting = “Merry #{holiday}, everyone”

long_greeting = <<END_STRING
    This is a long unquoted string
    which includes line breaks, formatting, etc
    We can also interpolate: #{greeting}
END_STRING
```

EXERCISE 4

courseware/exercises/4

Symbol

:there_can_be_only_one => *:there_can_be_only_one*

Evaluates to itself, unique

“Lonely enum value”

Hash

```
person = { "name" => "Tony S", :job => "waste management" }
person["name"] => "Tony S"
person[:job] => “waste management”
```

EXERCISE 5

courseware/exercises/5

I/O

```
puts “What’s your name?”
typed_name = gets

file = File.new(“test.txt”, “r”)
#... do stuff to file
file.close
```

better way, use a block:

```
File.open(“test.txt”, “w”) do |file|
    file.puts “Writing a line, then closing it”
end
```

EXERCISE 6

courseware/exercises/6

Environment

ARGV

ENV

EXERCISE 7

courseware/exercises/7

Regular Expressions

```
“can you find me?” =~ /me/              => 12
$1                                      => “me”
“can you find this?” =~ /not there/     => nil
cat_name = “Felix”
“Felix loves milk” ” =~ /${cat_name}/ => 0
“Not here” !~ /Felix/                    => true
```

Patterns

/^start of sentence/	*anchor: start of sentence*
/start of sentence$/	*anchor: end of sentence*
/[aeiou]/	*match characters a,e,i,o or u*
/[0-9]/	*match any number between 0-9*
/./	*match any character except whitespace*
/\d/	*match single digit (0-9)*
/\s/	*match whitespace char*
/\d*/	*match zero or more digits*
/\d+/	*match one or more digits*
/\d{1,3}/	*match number with length between 1 and 3*
/\d{3}/	*match digit exactly 3 numbers long*
/start (of the) sentence/	*matches and captures ("of the")*

EXERCISE 8

courseware/exercises/8

Shell Integration

```
`ls`      =>  <directory listing>
current_dir = `pwd`

file_path = “test.txt”
file_content = `pwd #{file_path}`
```

EXERCISE 9

courseware/exercises/9

Filesystem

EXERCISE 10

courseware/exercises/10

Your Turn!

EXERCISE 11

courseware/exercises/11

Wrapping Up

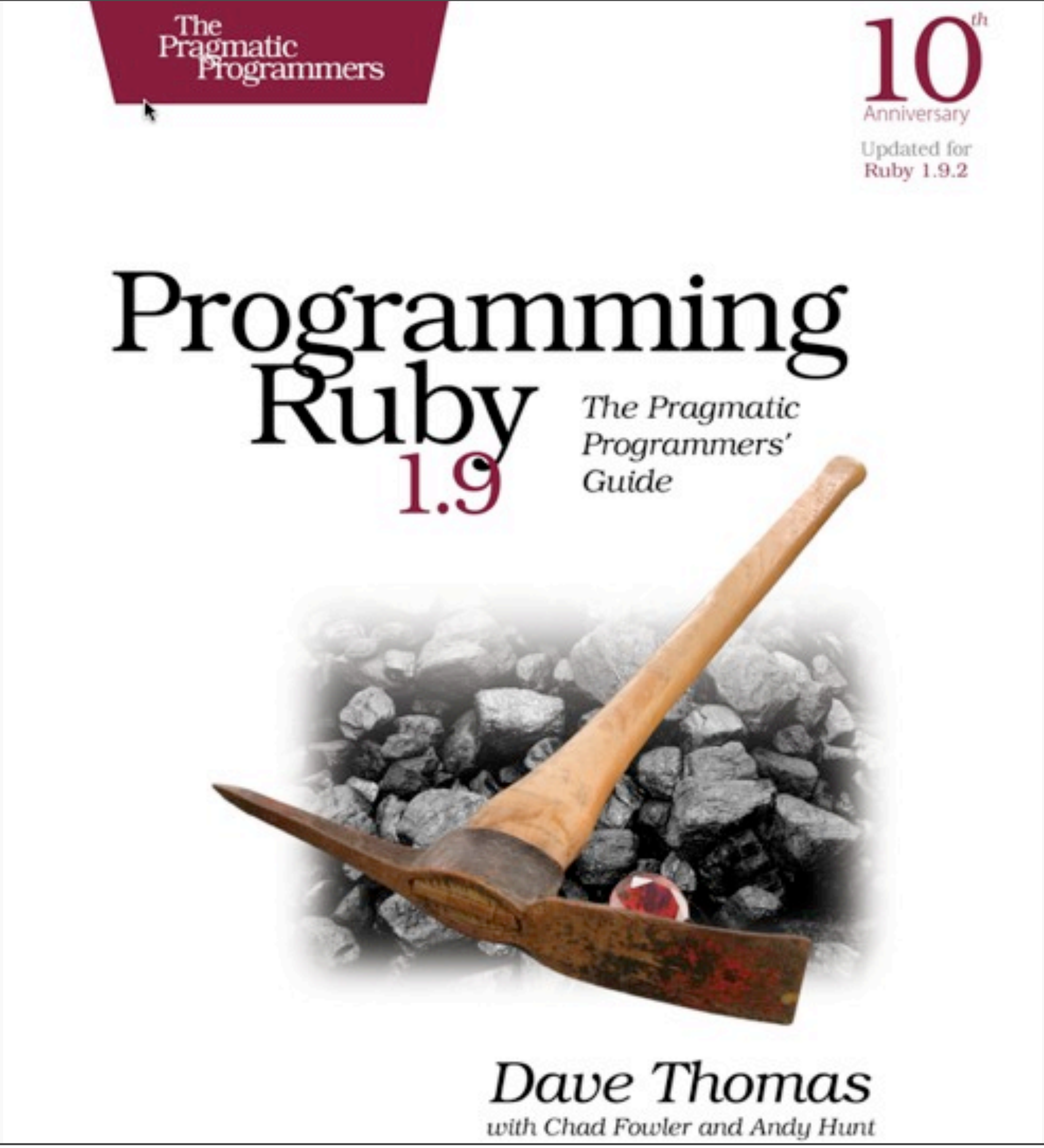
Huge Gaps!

what about classes, modules, etc etc?

no need for that yet

got enough for basic scripting!

References & Further Studies



The Pragmatic Programmers

10th Anniversary

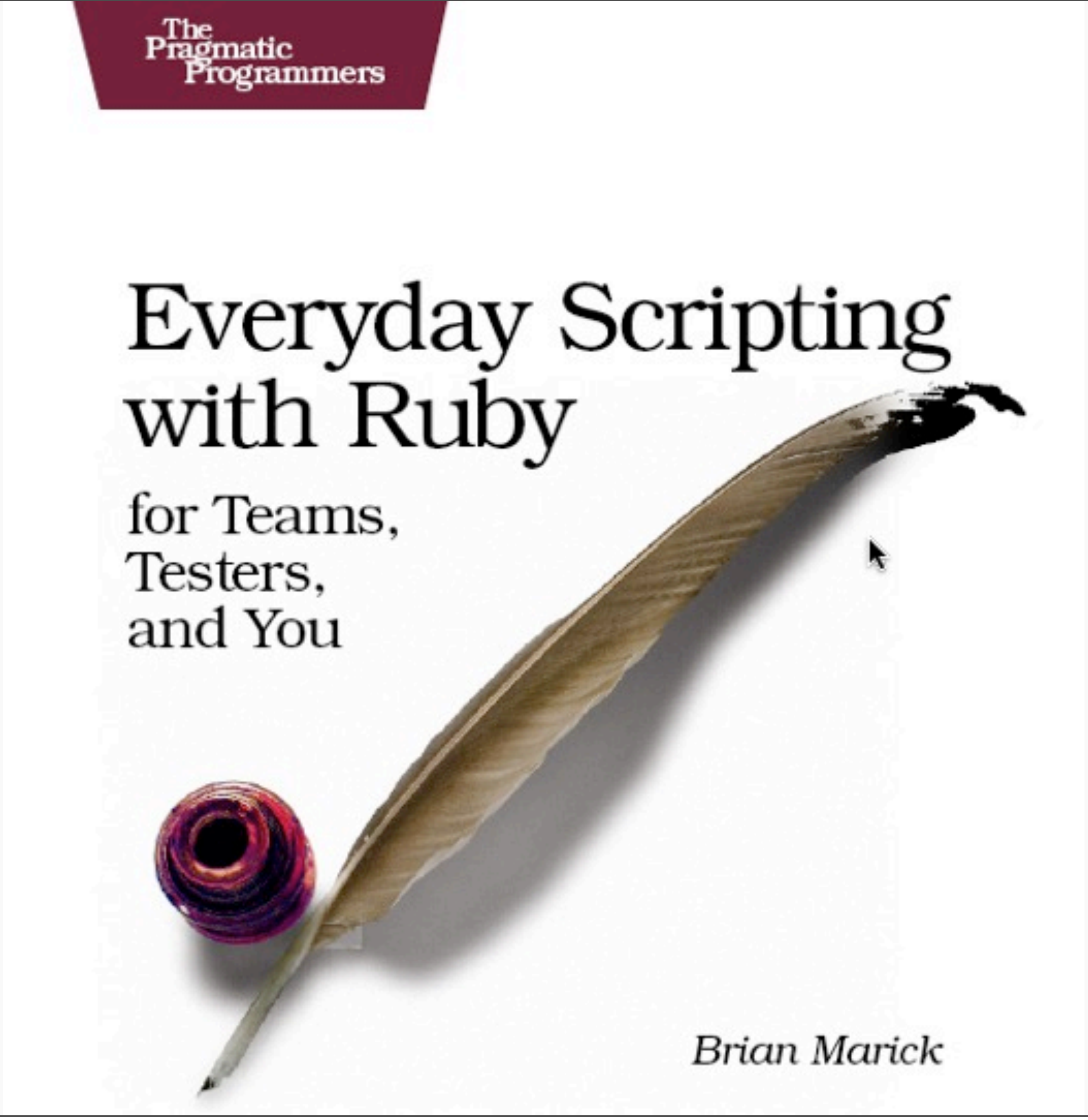
Updated for Ruby 1.9.2

Programming Ruby 1.9

The Pragmatic Programmers' Guide

Dave Thomas

with Chad Fowler and Andy Hunt



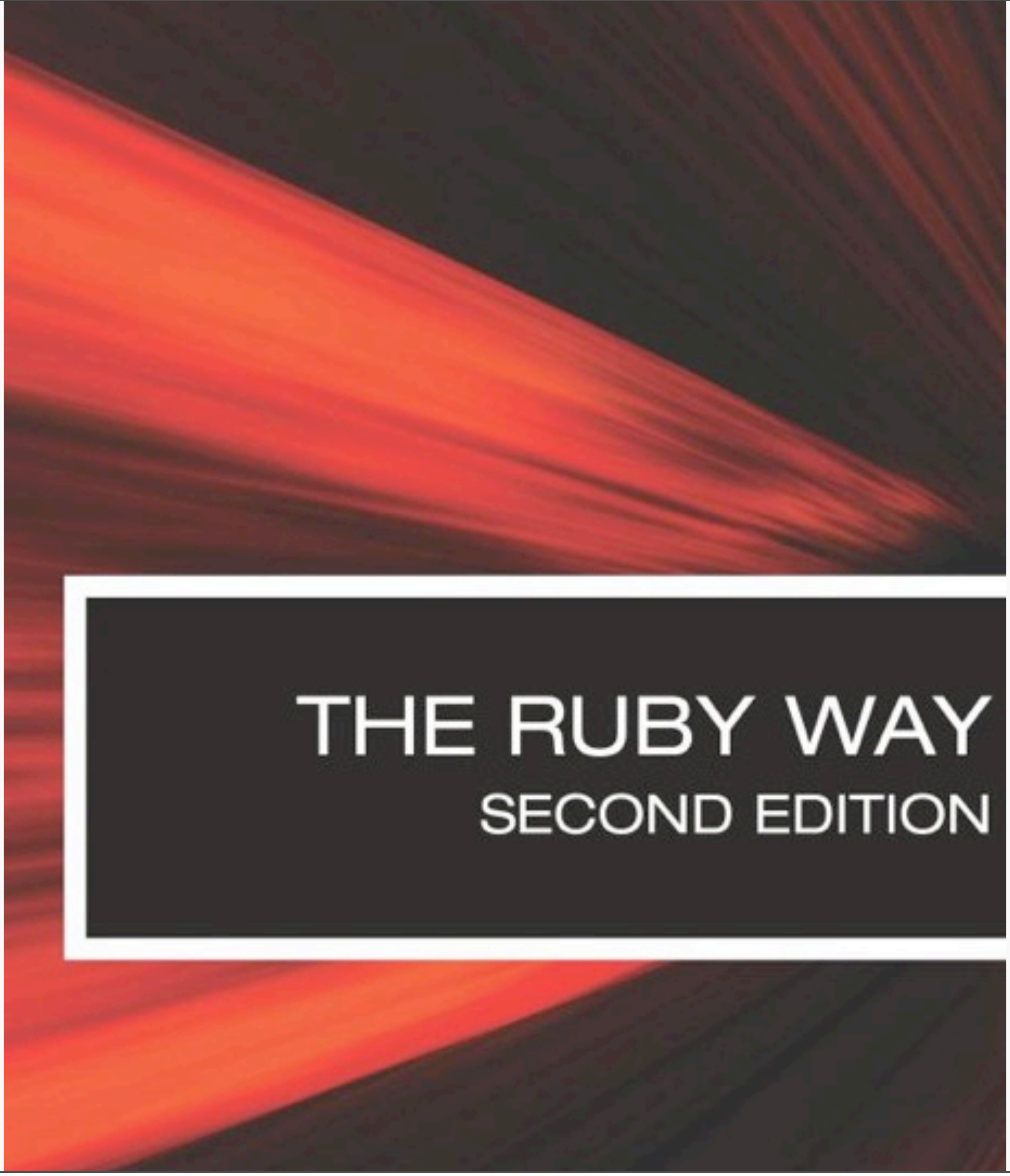
The Pragmatic Programmers

Everyday Scripting with Ruby

for Teams, Testers, and You

Brian Marick

Ruby
Best Practices


THE RUBY WAY
SECOND EDITION

Web Resources

http://www.ruby-lang.org/en/

http://www.rubyinside.com/

Contact Info

thomas@kjeldahlnilsson.net
linkedin.com/in/thomaskjeldahlnilsson
twitter.com/thomanil