Agile Webentwicklung mit Ruby on Rails

Prof. Dr. Oliver Vornberger
Nils Haldenwang, B.Sc.

UNIVERSITÄT OSNABRÜCK

Institut für Informatik
AG Medieninformatik

Einführung

Organisation, Überblick, Ablauf

Organisation

- Donnerstag: 10:15 - 11:45 Uhr, 31/449a
- Vorlesung mit integrierter Übung
- Übungsaufgaben freiwillig
- Scheinerwerb durch Programmierprojekt (3 LP)
- Website: http://www-lehre.inf.uos.de/~ror
- Mailingliste: ror12@list.serv.uni-osnabrueck.de

Agile Softwareentwicklung

- Menge von Methoden, Herangehensweisen und Prinzipien
- Vermeidung des “Big-Bang”
- inkrementelle und iterative Entwicklung
- Hier: Behaviour-Driven Development
- “State of the art” in Ruby Community

Ruby

- plattformunabhängige General Purpose Programmiersprache
- Veröffentlichung 1995 in Japan
- außerhalb Japans bekannt geworden um 2000
- Objektorientierung
- dynamische Typisierung (Duck Typing)
- Garbagecollection
- aktuelle Version: 1.9.3, veröffentlicht im Oktober 2011

Ruby

I hope to see Ruby help every programmer in the world to be productive, and to enjoy programming, and to be happy. That is the primary purpose of the Ruby language.

- *Yukihiro Matsumoto*

Ruby on Rails

- Webapplication Framework
- in Ruby geschrieben
- Open Source
- 2004 veröffentlicht
- aktuelle Version: 3.2, erschienen im Januar 2012

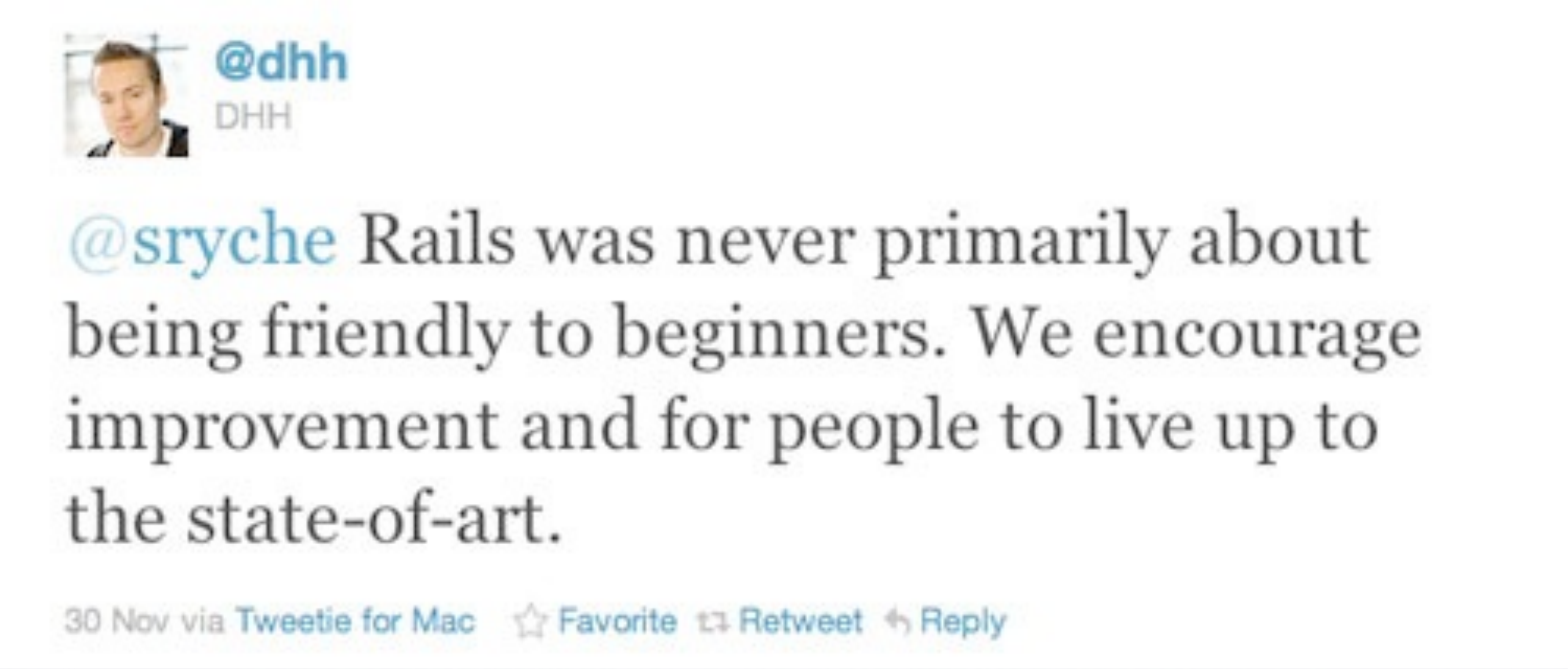
Ruby on Rails

Rails is about allowing beautiful code to solve the problems most people have most of the time in web-application development. It's about taking the pain away and making you happy.

This might all sound mighty fluffy, but only until you recognize that the single-most important factor in programmer productivity is motivation. And, happy programmers are certainly motivated programmers. Thus, if you optimize for happiness, you're optimizing for motivation, which ultimately leads to an optimization for productivity.

- David Heinemeier Hansson

Ruby on Rails



@dhh
DHH

@sryche Rails was never primarily about being friendly to beginners. We encourage improvement and for people to live up to the state-of-art.

30 Nov via Tweetie for Mac

Inhalt

- Umfassende Einführung in Ruby
- Versionskontrolle mit Git

- HTTP, REST und ROA
- Ruby on Rails: Einführung

- Behaviour-Driven Development
- Ruby on Rails: Vertiefung

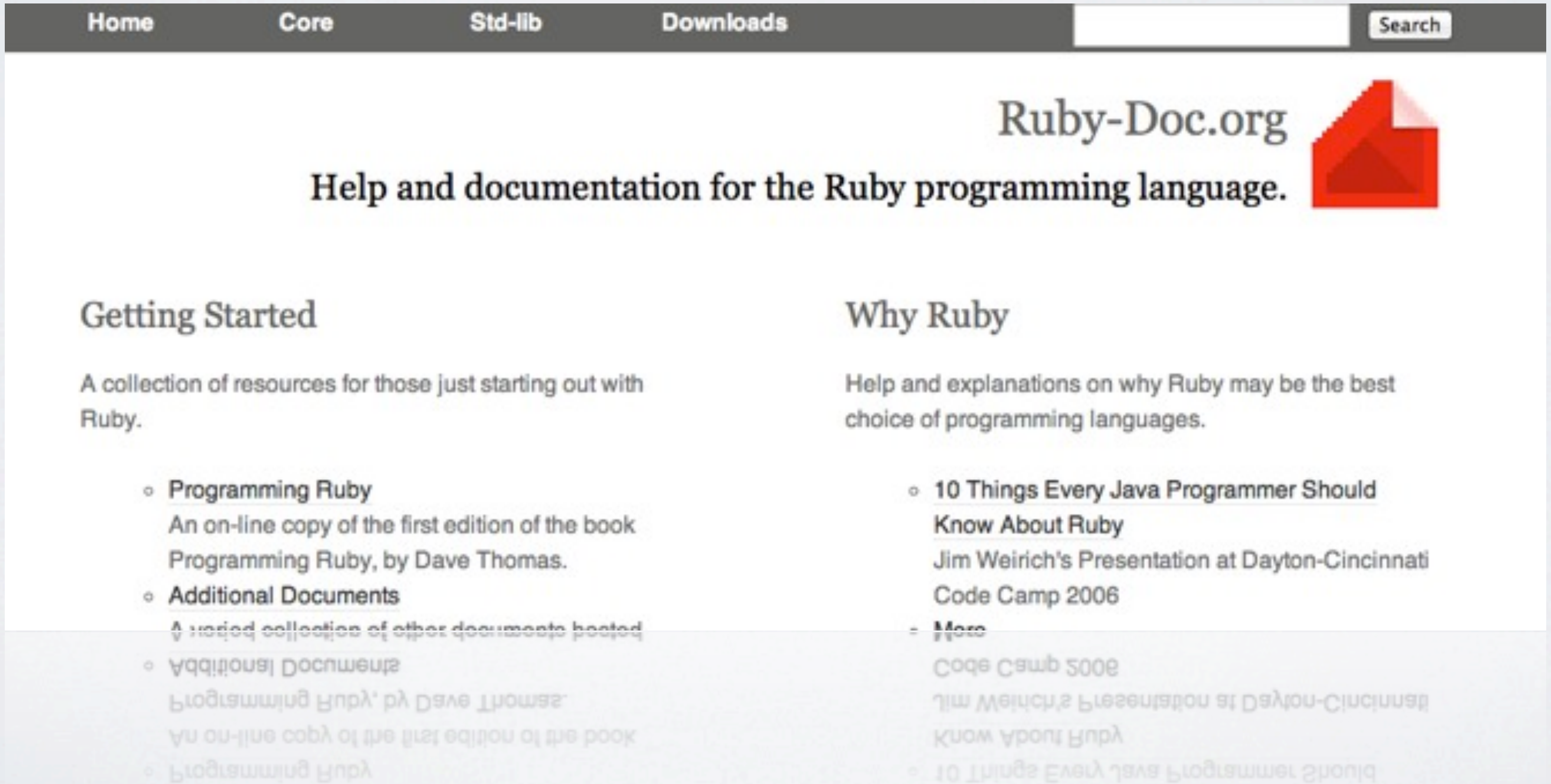
Ruby

Einführung in die Syntax

Quellen / Literaturempfehlungen

Ruby Dokumentation: http://www.ruby-doc.org/

1.9.3 Core API: http://www.ruby-doc.org/core-1.9.3/



Ruby-Doc.org

Help and documentation for the Ruby programming language.

Getting Started

A collection of resources for those just starting out with Ruby.

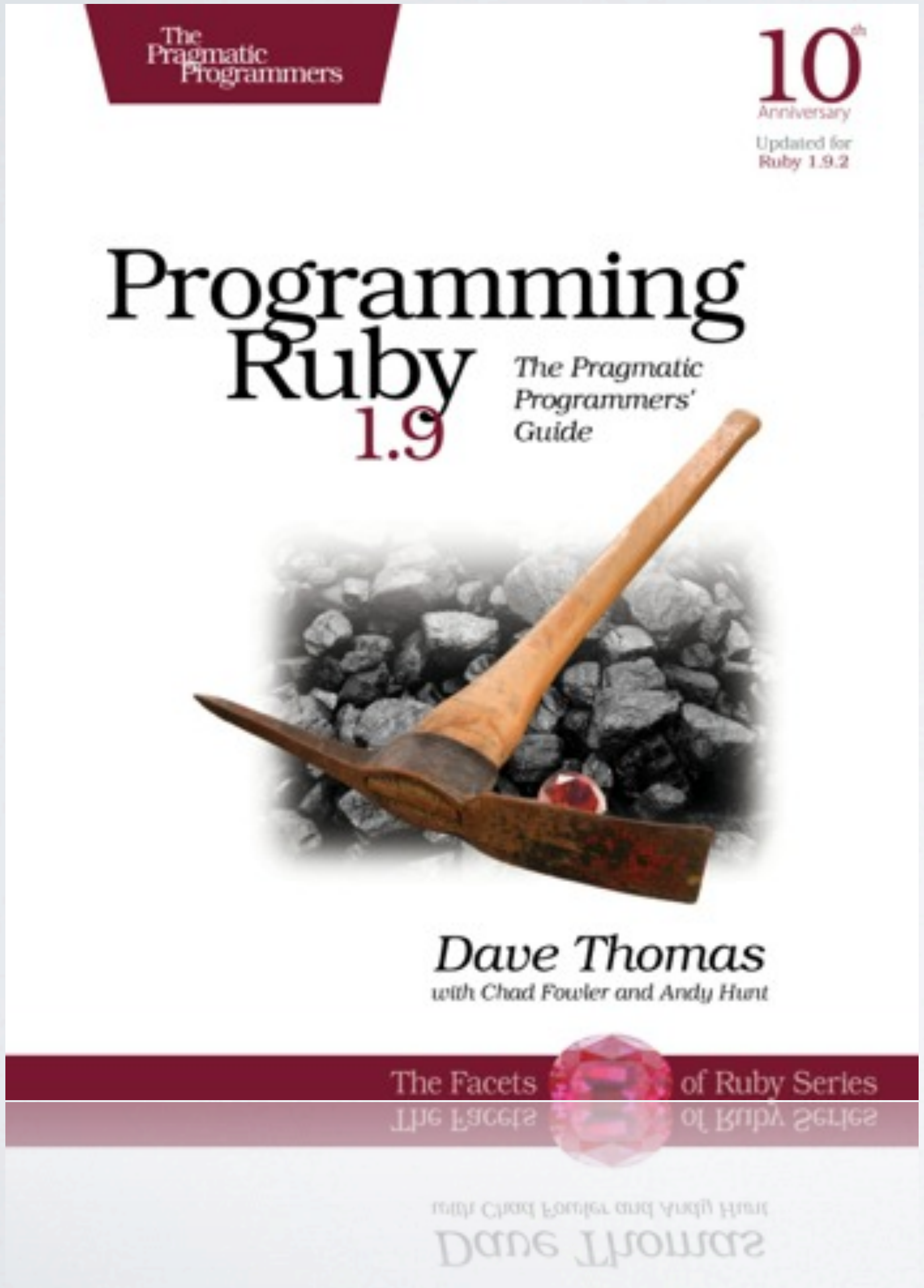
- Programming Ruby
  An on-line copy of the first edition of the book Programming Ruby, by Dave Thomas.
- Additional Documents
  A varied collection of other documents hosted

Why Ruby

Help and explanations on why Ruby may be the best choice of programming languages.

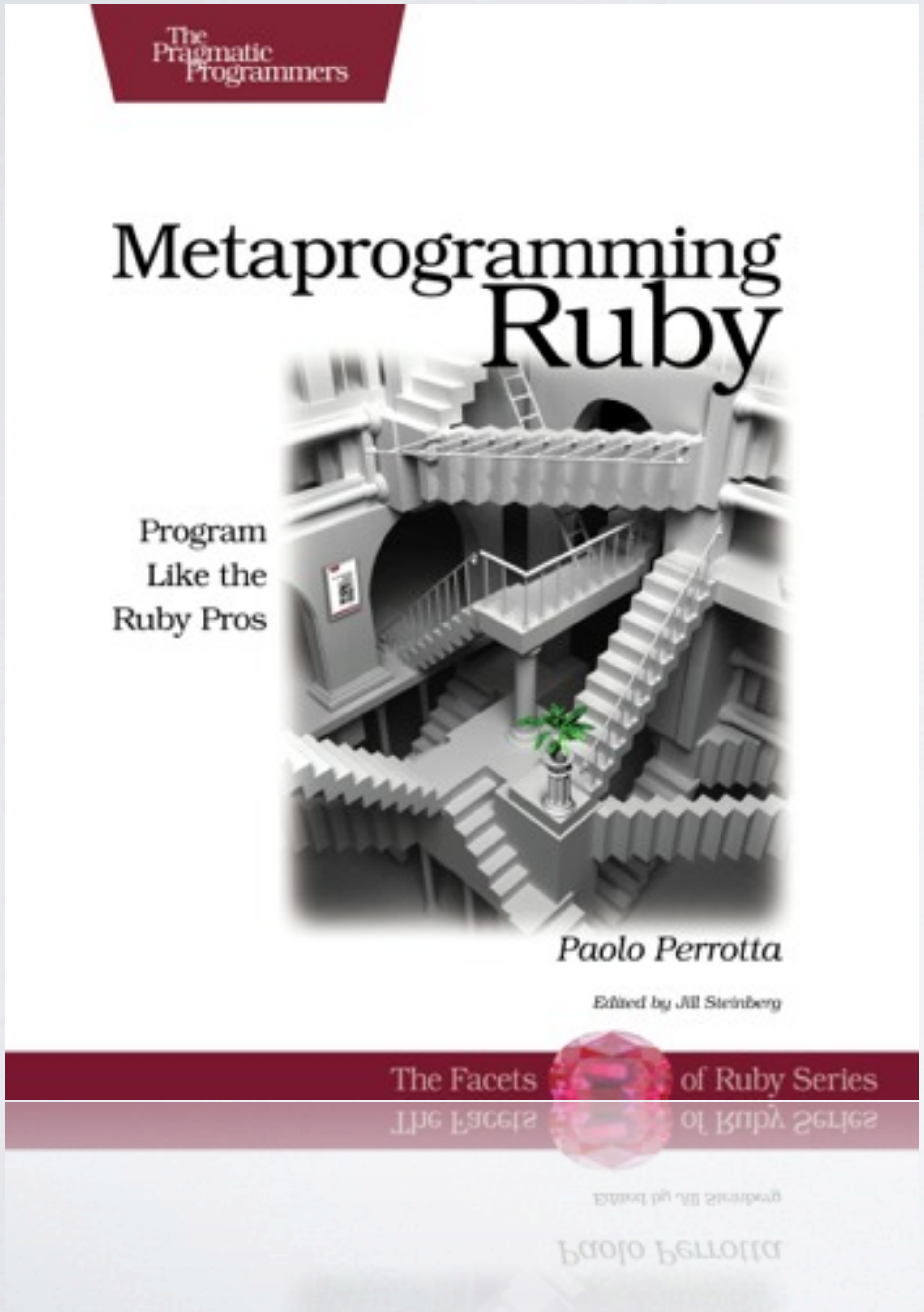
- 10 Things Every Java Programmer Should Know About Ruby
  Jim Weirich's Presentation at Dayton-Cincinnati Code Camp 2006
- More

Quellen / Literaturempfehlungen


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 Facets of Ruby Series

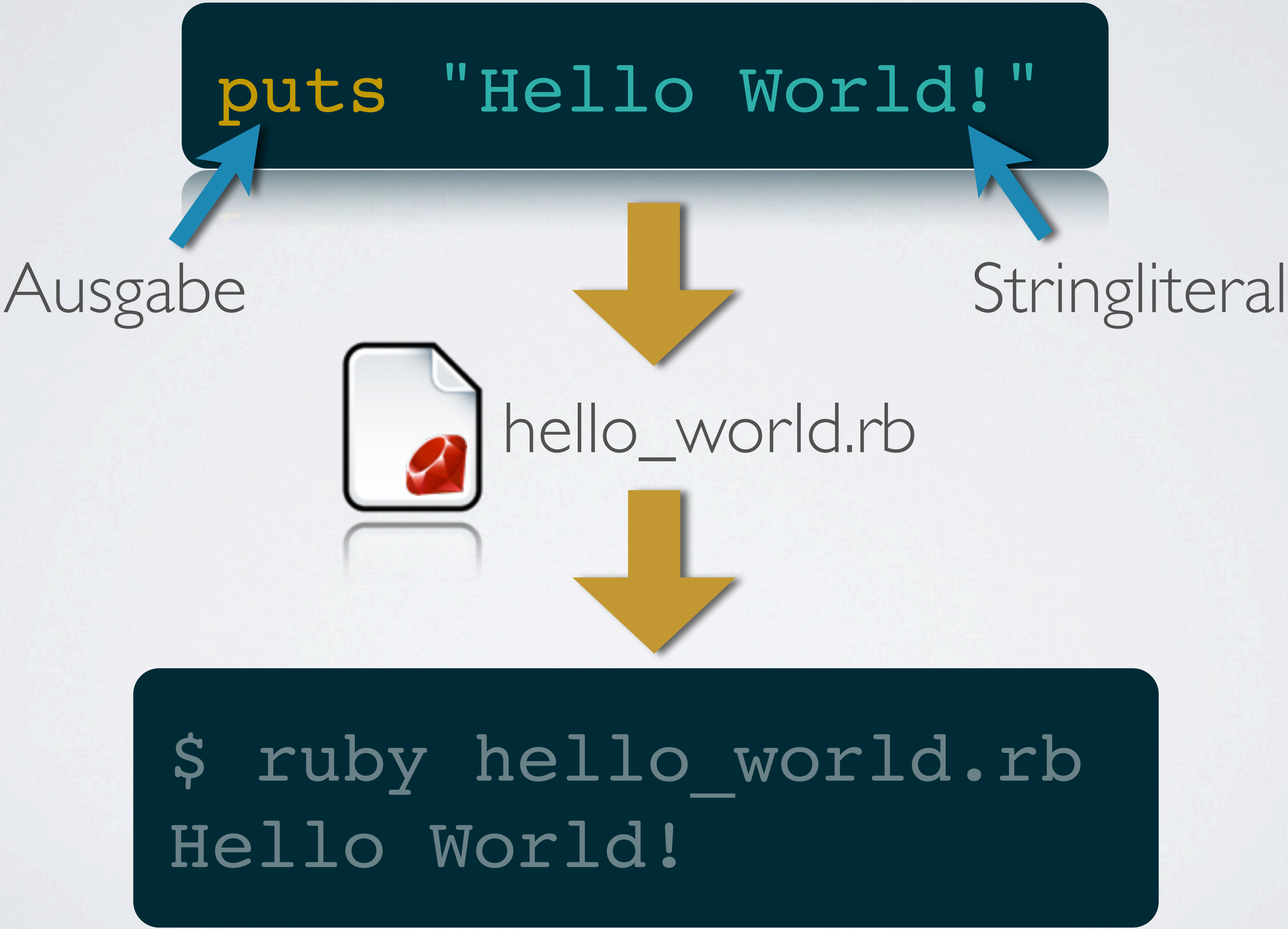
Dave Thomas
Chad Fowler,
Andy Hunt,
Programming Ruby 1.9
Pragmatic Bookshelf, 2009

Quellen / Literaturempfehlungen


The Pragmatic Programmers
Metaprogramming Ruby
Program Like the Ruby Pros
Paolo Perrotta
Edited by Jill Steinberg
The Facets of Ruby Series

Paolo Perrotta
Metaprogramming Ruby
Pragmatic Bookshelf, 2010

Hello World



```ruby
puts "Hello World!"
```

Ausgabe

Stringliteral

hello_world.rb

```
$ ruby hello_world.rb
Hello World!
```

Variablen

```
local_variable     = "Lokale Variable"
@instance_variable = "Instanzvariable"
@@class_variable   = "Klassenvariable"
$global_variable   = "Globale Variable"
SOME_CONSTANT      = "Konstante"
SomeClass          # Klassenkonstante
```

Basisdatentypen

```
natural_number = 42
natural_number.class # => Fixnum, konstante Zahl

float_number = 42.0
float_number.class # => Float, Fließkommazahl

string = "Hello World!"
string.class # => String

symbol = :hello_world
symbol.class # => Symbol, konstanter String
```

Collections: Array

```ruby
a = []
a = Array.new

a = [42, "foo", nil]

puts a[0]   # Ausgabe: 42

puts a.size # Ausgabe: 3

a[0] = 24
```

Collections: Hash

```ruby
h = Hash.new
h = {}

h = {
  :a  => "b",
  :c  => 42,
  "e" => "f",
  42  => "The answer,..."
} # 1.8 && 1.9

puts h[:a] # Ausgabe: b
puts h[42] # Ausgabe: The answer,..

h[:c] = 73

h = {
  a: "b",
  c: 42
} # Nur 1.9
```

Hashrocket wenn keine Symbole

Doppelpunkt bei Symbolen

Demo

Methoden

```ruby
def say_goodnight(name)
  result = "Good night, " + name
  return result
end

puts( say_goodnight("John-Boy") )
# => Good night, John-Boy

puts say_goodnight "John-Boy"
puts say_goodnight("John-Boy")

def say_goodnight(name)
  result = "Good night, #{name}"
  return result
end

def say_goodnight(name)
  "Good night, #{name}"
end
```

Klammern optional

Empfohlene Schreibweise

Rückgabewert ist der Wert des letzten Ausdruckes.

Bedingte Anweisungen

```ruby
if x == 5
  puts "x is 5"
end

puts "x is 5" if x == 5
```

```ruby
if x == 5
  puts "x is 5"
elsif x == 3
  puts "x is 3"
else
  puts "x is unknown"
end
```

Obacht!

Bedingte Anweisungen

```
if !( x == 5 )
  puts "x isn't 5"
end

puts "x isn't 5" if !( x == 5 )


unless x == 5
  puts "x is not 5"
end

puts "x is not 5" unless x == 5
```

Schleifen

```
i = 42
while i > 0
  puts i
  i -= 1
end

i = 42
until i == 0
  puts i
  i -= 1
end
```

```
a = [1, 2, 3, 4, 5]

for element in a
  puts element
end
```

Collatz

```
public class Collatz {
  public static int value(int x){

    int z = 0;

    while (x != 1) {

      if (x % 2 == 0)
        x = x / 2;
      else
        x = 3*x+1;

      z = z+1;
    }

    return z;
  }
}
```

Collatz

```java
public class Collatz {
  public static int value(int x){

    int z = 0;

    while (x != 1) {

      if (x % 2 == 0)
        x = x / 2;
      else
        x = 3*x+1;

      z = z+1;
    }

    return z;
  }
}
```

```ruby
def collatz(x)
  iterations = 0

  while (x != 1)

    if (x % 2 == 0)
      x = x / 2
    else
      x = 3*x + 1
    end

    iterations = iterations + 1
  end

  return iterations
end
```

Collatz

```ruby
def collatz(x)
  iterations = 0

  while (x != 1)

    if (x % 2 == 0)
      x = x / 2
    else
      x = 3*x + 1
    end

    iterations = iterations + 1
  end

  return iterations
end
```

```ruby
def collatz(x)
  iterations = 0

  until x == 1

    if x.even?
      x /= 2
    else
      x = 3*x + 1
    end

    iterations += 1
  end

  iterations
end
```

Blöcke

```ruby
{ puts "Hello, Block!" }

do
  puts "Hello, Block!"
  puts "Multiple lines."
end
```

Blöcke übergeben

```ruby
greet_block { puts "Hello, Block!" }

greet_block do
  puts "Hello, Block!"
end
```

Blöcke aufrufen

```ruby
def call_block
  puts "Calling block:"
  yield
  yield
  puts "Block has been called."
end

call_block { puts "Hello" }

# Ausgabe:
#   Calling block:
#   Hello
#   Hello
#   Block has been called.
```

Blöcke mit Parametern

```
def call_block
  yield("foo")
end

call_block { |param| puts "Param: #{param}" }

call_block do |param|
  puts "Param: #{param}"
end

# Ausgabe:
#   Param: foo
```

Variablen und Blöcke

```ruby
def call_block
  yield
end

var = 42

call_block { var = 73 }

puts var

# Ausgabe: 73
```

Optionale Blöcke

```
def greet(name)

  if block_given?
    yield name
  else
    puts "Hello, #{name}!"
  end

end

greet "Joe" # Ausgabe: Hello, Joe!

greet( "Joe" ) { |name| puts "Nice to meet you, #{name}!" }
# Ausgabe: Nice to meet you, Joe!
```

Iteratoren mit Blöcken

```ruby
a = ["iterators", "rock"]

a.each { |e| puts e }

# Ausgabe:
#   iterators
#   rock

h = { a: 42, b: 73 }

h.each { |k, v| puts "key: #{k}, value: #{v}" }

# Ausgabe:
#   key: a, value: 42
#   key: b, value: 73
```

Schleifen mit Blöcken

```ruby
3.times { |i| puts i }
# Ausgabe:
#   0
#   1
#   2

1.upto(3) { |i| puts i }
# Ausgabe:
#   1
#   2
#   3

3.downto(1) { |i| puts i }
# Ausgabe:
#   3
#   2
#   1
```

Demo

Ruby

Installation

Ruby installieren

Ruby Version Manager

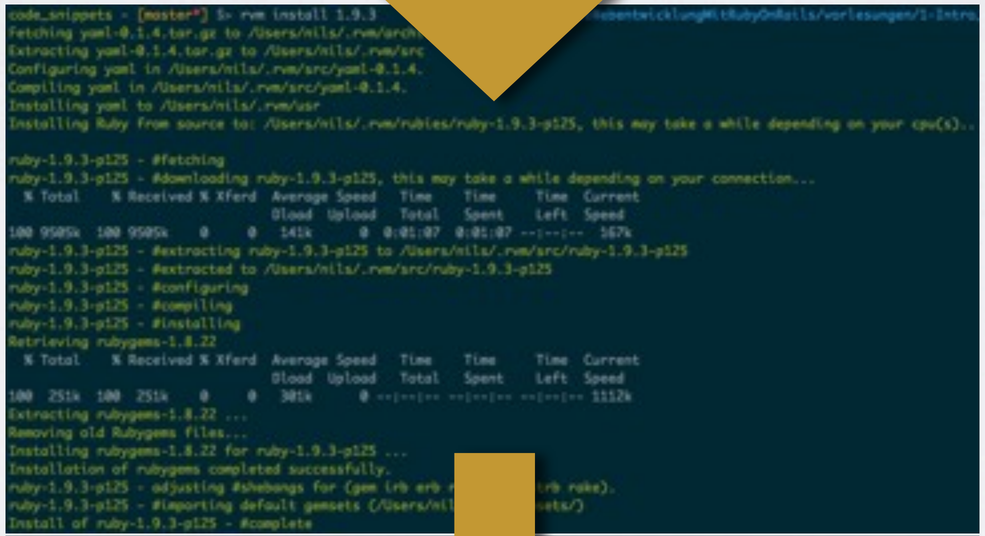
http://beginrescueend.com/

Ruby Installer

http://rubyinstaller.org/

Ruby installieren mit RVM

```
$ rvm install 1.9.3
```



```
code_snippets - [master*] $> rvm install 1.9.3                    oentwicklungMitRubyOnRails/vorlesungen/1-Intro
Fetching yaml-0.1.4.tar.gz to /Users/nils/.rvm/arch
Extracting yaml-0.1.4.tar.gz to /Users/nils/.rvm/src
Configuring yaml in /Users/nils/.rvm/src/yaml-0.1.4.
Compiling yaml in /Users/nils/.rvm/src/yaml-0.1.4.
Installing yaml to /Users/nils/.rvm/usr
Installing Ruby from source to: /Users/nils/.rvm/rubies/ruby-1.9.3-p125, this may take a while depending on your cpu(s)..

ruby-1.9.3-p125 - #fetching
ruby-1.9.3-p125 - #downloading ruby-1.9.3-p125, this may take a while depending on your connection...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 9505k  100 9505k    0     0   141k      0  0:01:07  0:01:07 --:--:--  167k
ruby-1.9.3-p125 - #extracting ruby-1.9.3-p125 to /Users/nils/.rvm/src/ruby-1.9.3-p125
ruby-1.9.3-p125 - #extracted to /Users/nils/.rvm/src/ruby-1.9.3-p125
ruby-1.9.3-p125 - #configuring
ruby-1.9.3-p125 - #compiling
ruby-1.9.3-p125 - #installing
Retrieving rubygems-1.8.22
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  251k  100  251k    0     0   301k      0 --:--:-- --:--:-- --:--:-- 1112k
Extracting rubygems-1.8.22 ...
Removing old Rubygems files...
Installing rubygems-1.8.22 for ruby-1.9.3-p125 ...
Installation of rubygems completed successfully.
ruby-1.9.3-p125 - adjusting #shebangs for (gem irb erb r          irb rake).
ruby-1.9.3-p125 - #importing default gemsets (/Users/nil          sets/)
Install of ruby-1.9.3-p125 - #complete
```

```
$ rvm use 1.9.3
$ irb
1.9.3-p0 >
```

Fragen, Kritik, Wünsche oder Anregungen?