



RUBY.rb [2.6.5]

GLIEDERUNG

- 3. Fakten**
- 4. „Hello World“**
- 5. Zahlen**
- 6. Strings**
- 7. Arrays und Hashes**
- 8. Namen und Symbole**
- 9. Bedingungen**
- 10. Schleifen**
- 11. Methoden**
- 12. Klassen und Instanzen**
- 13. Vererbung**
- 14. Module**
- 15. Ruby on Rails**
- 16. Quellen**

FAKTEN [1995, Yukihiro „Matz“ Matsumoto]

- + open source
- + Objektorientiert
- + reich an Bibliotheken > Ruby on Rails
- + entwicklerfreundliche Sprache
- + Ruby-Interpreter nötig

```
1 class Testklasse
2   def hallosagen(name)
3     puts "Hallo #{name}!"
4   end
5 end
6
7 testkl = Testklasse.new
8
9 testkl.hallosagen("Anton")
```

<https://www.ruby-lang.org/en/downloads>

apt-get install ruby

GITLAB-HDM:/am180/ruby

„HELLO WORLD“

puts

- Ausgabe der Zeichenkette innerhalb den Anführungszeichen

p

- Kurze Schreibweise
- Ausgabe in Anführungszeichen

print

- Ausgabe der Zeichenkette ohne Zeilenumbruch „\n“

```
1 puts "Hello World!"
2
3 print "Ausgabe ohne Zeilenumbruch - "
4
5 p "Die kürzere Variante"
6
7 =begin
8 => Hello World
9 => Ausgabe ohne Zeilenumbruch - "Die
10 kürzere Variante"
11 =end
```

ZAHLEN

Objekte des Types

- FixNum
-2³⁰ bis +2³⁰ [32bit Maschine]
- BigNum
-2⁶² bis +2⁶² [64bit Maschine]

Arithmetische Operatoren

- „+“ Addition
- „-“ Subtraktion
- „*“ Multiplikation
- „/“ Division
- „%“ Modulo, Division mit Rest
- „**“ Potenzfunktion
- „+=“ Hochzählen
- „-=“ Herunterzählen

```
1 a = 7 # FixNum
2 b = 3
3 bignum = 123456764389012345678904351234567890 # BigNum
4
5 addition = a+b
6 subtraktion = a-b
7 multiplikation = a*b
8 division = a/b
9 modulo = a%b
10 potenzfunktion = a**b
11
12 puts a+=2 # Operatoren ++ und -- gibt es nicht
13
14 puts addition, subtraktion, multiplikation
15 puts division, modulo, potenzfunktion
16 puts "-----"
17
18 b.upto(a) { |n| puts n } # Methoden der Basisklasse Integer
19 puts "-----"
20 a.downto(b) { |n| puts n }
```

STRINGS

erzeugt durch

- einfache Hochkommata oder
- doppelte Hochkommata oder
- %q oder %Q

Ausdrücken **`#{Ausdruck}`**
können in einem String
nur innerhalb
doppelter Hochkommata
definiert werden.

```
1 mystr = 'Meine Zeichenkette'
2 mystr2 = %q{Mein Text 2}
3 myname = 'Anton'
4 salut = "Hallo #{myname}"
5 mystr3 = %Q{Hallo #{myname}}
6
7 puts mystr
8 puts mystr2
9 puts mystr3
10 puts salut
11
12 =begin
13 Konsolenausgabe:
14 => Meine Zeichenkette
15 => Mein Text 2
16 => Hallo Anton
17 => Hallo Anton
18 =end
```

ARRAYS UND HASHES

Arrays

- Typ der Klasse *Array*
- Erzeugung mittels `[]`
- alternativ über **new** erzeugbar
- in der Länge (`.length`) unbegrenzt
- bei Bedarf automatisch erweitert
- Indizierung beginnt bei 0

Hashes

- Typ der Klasse *Hash*
- ein Paar aus Schlüssel und Wert
- Erzeugung mittels `{}`
- alternativ über **new** erzeugbar

```
1 myarray = [21,5,"Testzeichen"]
2 myarray[3] = 243
3 x = 1
4
5 for i in myarray
6     if x != myarray.length then print "#{i} - "
7     else print "#{i}"
8     end
9     x += 1
10 end
11
12 myHash = Hash.new()
13 myHash["Key1"] = "Wert1"
14 myHash[:Key2] = "Wert2"
15
16 puts myHash["Key1"]
17 puts myHash[:Key2]
```


NAMEN UND SYMBOLE

Beginn: Kleinbuchstaben / Unterstrich

- lokale Variable
- Methodenparameter

Beginn: Großbuchstaben

- Klassennamen
- Modulnamen
- Konstanten

Beginn: Klammeraffen @

- Klassenvariablen
- Instanzvariablen

Beginn: Dollarzeichen \$

- globale Variablen

```
1 string = "Thomas"
2 string1 = "Thomas"
3 symbol1 = : "Ralf"
4 symbol2 = : "Ralf"
5
6 puts string.object_id, string1.object_id
7 puts "-----"
8 puts symbol1.object_id, symbol2.object_id
9
10 =begin
11 Konsolenausgabe:
12 => 39393320
13 => 39393300
14 => -----
15 => 1102948
16 => 1102948
17 =end
```


BEDINGUNGEN

Jede **if**-Anweisung ist mit einem **end**

Operatoren für die Ausdrücke

<code>==</code>	<code>!=</code>
<code><</code>	<code><=</code>
<code>></code>	<code>>=</code>

Ausdrücke verknüpfen

<code>&&</code>	und
<code> </code>	oder

Statt die Bedingung einer **if**-Anweisung durch **!** zu negieren, bietet Ruby die **unless**-Anweisung

```
1 x = 5
2 if x == 0
3   puts "0"
4 elsif x < 0
5   puts "kleiner 0"
6 else
7   puts "größer 0"
8 end
9
10 if x == 0 then puts "0"
11   elsif x > 0 then puts "kleiner 0"
12   else puts "größer 0"
13 end
14
15 puts x if x > 6      # Kurzzeiler
16 puts "x ist kleiner wie 6" unless x > 6
17
18 =begin
19 Konsolenausgabe:
20 => größer 0
21 => kleiner 0
22 => x ist kleiner wie 6
23 =end
```

SCHLEIFEN

for-Schleife

- iteriert über die Elemente eines Bereichs
- Arrays oder auch Hashes

while-Schleife

- Block wird ausgeführt, solange Bedingung **true** ist

until-Schleife

- Block wird ausgeführt, solange Bedingung **false** ist

```
1  s1 = [1,2,3,4,5,6]
2  s2 = 5
3  s3 = 0
4
5  for i in s1
6    puts i
7  end
8
9  puts "-----"
10
11 while s2 > 1 do
12   puts s2 -= 1
13 end
14
15 puts "-----"
16
17 until s3 >= 4 do
18   puts s3 += 1
19 end
```

METHODEN

Typangabe entfällt

- für Parameter und Rückgabewerte

return kann entfallen

- können mehr als einen Rückgabewert haben

Überladen

- umgehen durch unterschiedliche Zahl von Argumenten
- Defaultbelegung von Argumenten möglich

```
1 def vorstellen(*args)
2   case args.size
3     when 0
4       puts "Nix bekommen!"
5     when 1
6       puts "Hallo #{args[0]}!"
7     when 2
8       puts "#{args[0]} ist #{args[1]} Jahre alt!"
9     when 3
10      puts "#{args[0]} ist #{args[1]} Jahre alt und lebt in #{args[2]}"
11   end
12 end
13
14 vorstellen
15 vorstellen "Anton"           # möglich, aber ggf. Warnung
16 vorstellen("Dana", 21)       # eindeutig und besser
17 vorstellen("Dana", 21, "Stuttgart")
18
```

KLASSEN UND INSTANZEN

Klassendefinition

- *class Classname*
end

Erzeugung einer Instanz

- Aufruf der Methode `new`

Instanzvariablen

- innerhalb einer Instanzmethode

Klassenmethoden

- per Default öffentlich

```
1  class Student
2      attr_reader :semester
3      attr_writer :semester
4
5      def initialize(name, vorname, fachrichtung, semester=1)
6          @name = name          # Instanzvariable
7          @vorname = vorname
8          @fachrichtung = fachrichtung
9          @semester = semester
10     end
11 end
12
13 teststudent = Student.new("Mustermann", "Max", "MI-7")
14
15 puts teststudent.semester
```

VERERBUNG

Subklasse erbt

- Variablen
- Konstanten
- Methoden (auch *initialize*)

Subklasse hat Zugriff

- Methoden der Superklasse
 - auch private
 - Kann die Sichtbarkeit für eigene Instanzen anpassen

```
1  require './12_klassen_und_instanzen'
2
3  class Student < Person
4      attr_reader :semester
5      attr_writer :semester
6
7      def statusinfo
8          "Status: #{@status}"
9      end
10 end
11
12 teststudent = Student.new("Mustermann", "Max", "Student", "mm123")
13
14 puts teststudent.statusinfo
```


MODULE

Mehrfach in Klassen verwendbar

- Aufruf der Methode new

Vorgehensweise

- Module in die Klasse integrieren
- Methoden via Instanzen nutzen

```
1 module TestModule1
2   def fnc1
3     "Testmodule 1"
4   end
5 end
6 module TestModule2
7   def fnc2
8     "Testmodule 2"
9   end
10 end
11
12 class Testklasse
13   attr_reader :name
14   include TestModule1
15   include TestModule2
16
17   def initialize(name)
18     @name = name
19   end
20 end
21 testklasseninstanz = Testklasse.new("Max Mustermann")
22
23 puts testklasseninstanz.name
24 puts testklasseninstanz.fnc1, testklasseninstanz.fnc2
```


RUBY ON RAILS [2005, David Heinemeier Hansson]

Framework für Web-Anwendungen

- Aufruf der Methode new

Interesse geht deutlich zurück!

- wenige Fragen auf Stackoverflow
- Wandlung der Web-Entwicklung
- Erfolg von Node.js



QUELLEN

<https://www.ruby-lang.org/de/downloads/>

<https://www.digitalocean.com/community/tutorials/how-to-write-your-first-ruby-program>

http://openbook.rheinwerk-verlag.de/ruby_on_rails/

<https://jaxenter.de/einfuehrung-und-uberblick-in-die-webentwicklung-mit-ruby-on-rails-9479>

https://www.bader-jene.de/wp-content/uploads/Einfuehrung_Ruby_on_Rails.pdf

https://guides.rubyonrails.org/getting_started.html

<https://www.javatpoint.com/ruby-on-rails-introduction>

<http://railsapps.github.io/what-is-ruby-rails.html>

<https://www.tutorialspoint.com/ruby-on-rails/rails-introduction.htm>

<https://www.learnenough.com/ruby-on-rails-6th-edition-tutorial>

<http://b-simple.de/download/ruby.pdf>

<https://redmonk.com/sogrady/2019/03/20/language-rankings-1-19/>

<https://t3n.de/news/ruby-on-rails-interesse-842348/>

<https://syndicode.com/2018/11/01/alles-uber-ruby-werkzeuge-besonderheiten-und-tipps/>

<https://naturaily.com/blog/who-gives-f-about-rails>