Laboratorio di Architetture Software e Sicurezza Informatica [AAF1569]

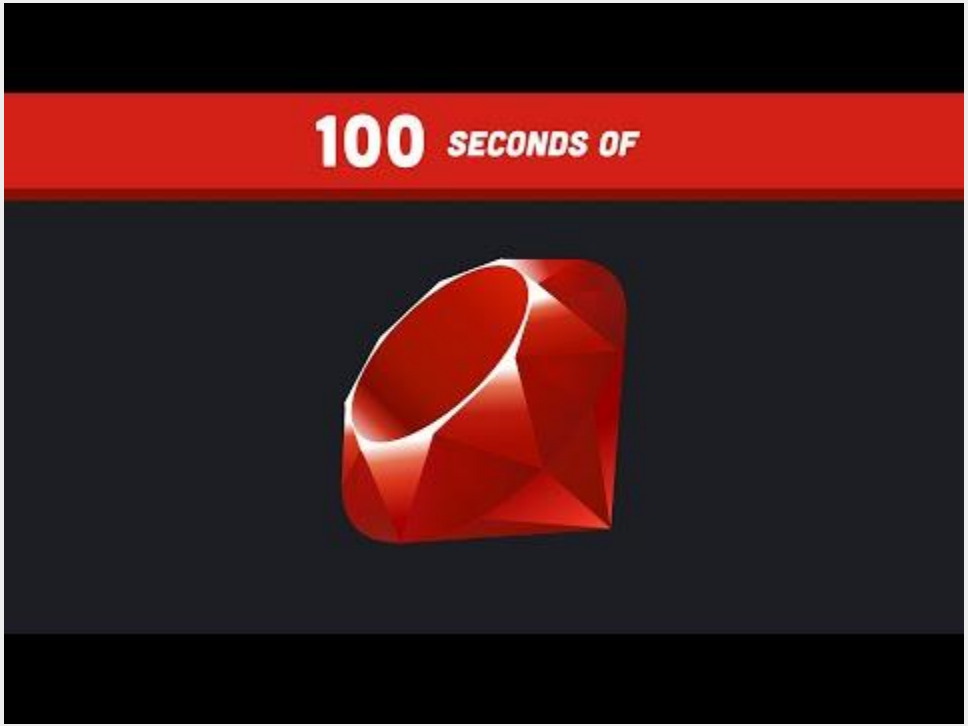
2 - Ruby Essentials

SAPIENZA
UNIVERSITÀ DI ROMA

Part of therein contents are based on slides of Prof. John Yang

DIAG
Dipartimento di Ingegneria informatica, automatica e gestionale Antonio Ruberti

Ruby in 100 seconds



https://www.youtube.com/watch?v=UYm0kfnRTJk

Ruby

A minimalistic language with rich libraries and few mechanisms.

Principles:

1. **Interpreted**
2. **Dynamically Typed**
3. OO - **Everything is an object**
    a. -> every operation is a method call
    b. -> every method call returns a value (== an object)
    c. -> every statement is an expression that returns a value (== an object)
4. Accesses to instance variables outside a class take place via accessor methods
    a. instance variables lacking public accessor methods are effectively private

Installation and Tools

https://www.ruby-lang.org/it/documentation/installation/

Additional tools:

- **RVM** for Linux & MacOS: https://rvm.io/rvm/install
  (see also https://nrogap.medium.com/install-rvm-in-macos-step-by-step-d3b3c236953b)
- RubyInstaller for Windows: https://rubyinstaller.org/
- WSL for Windows: https://docs.microsoft.com/it-it/windows/wsl/install
- Visual Studio Code https://code.visualstudio.com/ + Ruby extensions
- RubyMine IDE: https://www.jetbrains.com/ruby/

Getting Started with Ruby

1. Check your installation
   `ruby --version`
2. Use the Ruby interpreter
   `ruby hello_world.rb`
3. Interactive Ruby (irb) console
   `irb`

```
puts “Hello World by Ruby!”
```

Data Types

Everything is an object

- Numeric
- Boolean
- String
- ...

Comments

```
# single line comment


=begin
   multi-line comment
=end
```

Note: `=begin` and `=end` cannot be indented

if & the rest

```
if conditional [then]
   code...
[elsif conditional [then]
   code...]...
[else
   code...]
end

&& # and
|| # or
!  # not

puts "you are still young" if age < 105
# inline if
```

Allowed Boolean values: **`true`** and **`false`**

A non-boolean value that counts as true is called "**truthy**," and a non-boolean value that counts as false is called "**falsey**."

`nil` is evaluated as **falsey**

All other values (e.g., number *zero*, the empty string, the empty array, and so forth) are **truthy**.

`nil` may signal either Boolean falseness or a variable that refers to nothing. Used as result of an operation that otherwise would yield no meaningful return value

Types of Variables

```
local_variables

@instance_variable

@@class_variable

ClassName

Constant/CONSTANT

$global_variable
```

foo, @foo, @@foo, FOO, Foo, $FOO are all distinct

Classes

```
class ClassName
    @@class_variable_default = “something_default_shared”
    CONSTANTVAR = ‘192.168.0.1’
    def initialize(first_arg, second_arg = 0, *other_args)
        @instance_var1 = first_arg
        @instance_var2 = sec_arg
    end
    # metaprogramming
    attr_accessor :instance_var1  # can read and write this attribute
    attr_reader :instance_var2    # can only read this attribute
end
```

Class code examples

Link: https://github.com/dcdelia/lab-assi/tree/2023-24/ruby-intro/examples

1. Alike to the one we just discussed
2. Verbose vs. concise declaration of getter/setter/initialization methods
3. Subclasses and getter/setter methods
4. Subclasses and polymorphism
5. Side-effects from a method
6. Variable-length argument list

Check Instance Class

`Object.class()` method

`Object.is_a?(Class_name)` method

Methods Extra

Method names may end with a `!` , `?` or `=`

```
def title=(new_title)      # assignment method (note: it will always return its arguments)

def is_year_leap?()        # returns a boolean

def capitalize!()          # side-effect on object called on
```

All methods return a value. In the lack of an explicit return statement, the value of the last evaluated expression becomes the method return value.

Arrays

https://ruby-doc.org/core-3.1.1/Array.html

Array indexing starts at 0

Square brackets [] to declare one

Comma-separated elements. Multiple types supported within an array

Blocks

Literally, "blocks" of code that can be passed and called

Defined through curly braces { } or **do-end** sequences

```
[1, 2, 3].each { |n| puts "Current number is: #{n}" }
[1, 2, 3].each do |n|
  text = "Current number is: #{n}"
  puts text
end
```

Range

A range is a collection of values that are between the given begin & end values

```
(1..4).to_a# => [1, 2, 3, 4]

('a'..'d').to_a

    # => ["a", "b", "c", "d"]

(1...4).to_a       # => [1, 2, 3]

r = (..4)   # => nil..4, beginless

r = (1..)     # => 1.., endless

r.include?(-50)# => true
```

```
a = []

r = (1..)

r.each do |i|

  a.push(i) if i.even?

  break if i > 10

end
```

https://ruby-doc.org/core-3.1.1/Range.html

Loops

```ruby
for i in 1...10
    puts i
end

3.times do
    puts "Hello!"
end

a = ['gio', 'nick', 'ale']
for i in a
    puts i
end
```

```ruby
i = 0; t = 3

while i<t  do

    puts("Iteration i = #{i}" )

    i +=1

    if i >= t; break; end

end
```

https://www.tutorialspoint.com/ruby/ruby_loops.htm

Hashes

Hashes are what other languages call associative arrays or hashmaps

https://ruby-doc.org/core-3.1.1/Hash.html

```
h = {foo: 0, :bar => 1, 'baz': 2, '1':3}
h[:foo]
h[:foo] = 2
h.delete(:foo)
h = {}
```

Regular Expressions

https://ruby-doc.org/core-3.1.1/Regexp.html

```
'haystack' =~ /hay/      # position of (first) match, nil otherwise

/a+/.match('haystack')   # MatchData, nil otherwise
```

Regular expressions are usually delimited with forward slashes. If a string contains the pattern, it is said to match.

Extras

Ruby allows **omitting parentheses around argument lists** when doing so does not result in ambiguous parsing (“poetry mode”)

When the last argument to a method call is a hash, the curly braces for the hash literal can be omitted: `foo(arg, key: val)` vs. `foo(arg, {key: val})`

Sugared	De-sugared
`10 % 3`	`10.modulo(3)`
`5+3`	`5.+(3)`
`x == y`	`x.==(y)`
`a * x + y`	`a.*(x).+(y)`
`a + x * y`	`a.+(x.*(y))`
`x[3]`	`x.[](3)`
`x[3] = 'a'`	`x.[]=(3,'a')`

Hands-on

Lab taken from the SaaS book in use to this course:

https://github.com/dcdelia/lab-assi/tree/2023-24/ruby-intro/book-lab