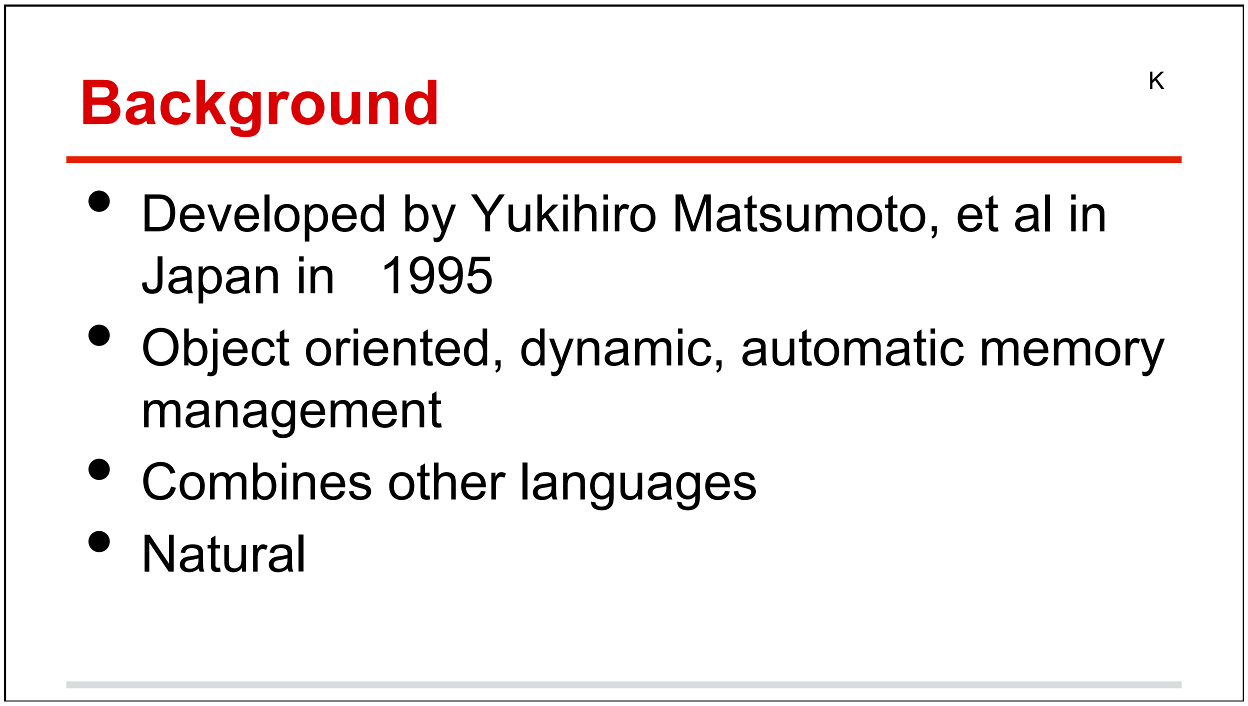
Ruby

Andrew McCurdy & Kevin Glasshoff



Background

K

- Developed by Yukihiro Matsumoto, et al in Japan in 1995
- Object oriented, dynamic, automatic memory management
- Combines other languages
- Natural

Sectors of Use

A

3D Modeling: Google SketchUp

Networking: ODS

Web Applications

Rails

K

- Rails is an open source web development framework for Ruby.
- Originally developed by David Hansson.
- Used in most large/enterprise Ruby projects that are designed to run on the web.
- Sinatra
- Padrino

Object Oriented Features

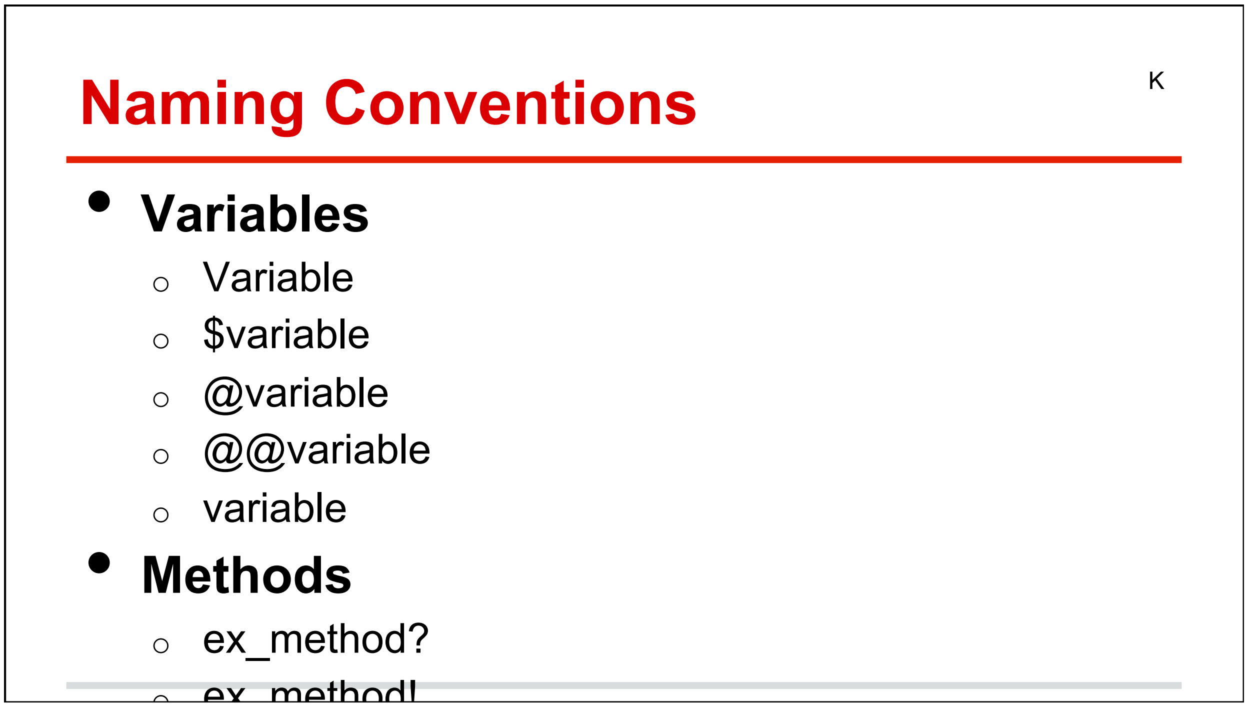
K

- Ruby is a pure object-oriented language i.e. everything in it is an object.
- Classes, attributes, methods, inheritance…
- ‘<’ signifies inheritance from another class
- CANNOT inherit from multiple parents, but can have multiple “Mixins”.
  - Modules

Object Oriented Features (cont.)

K

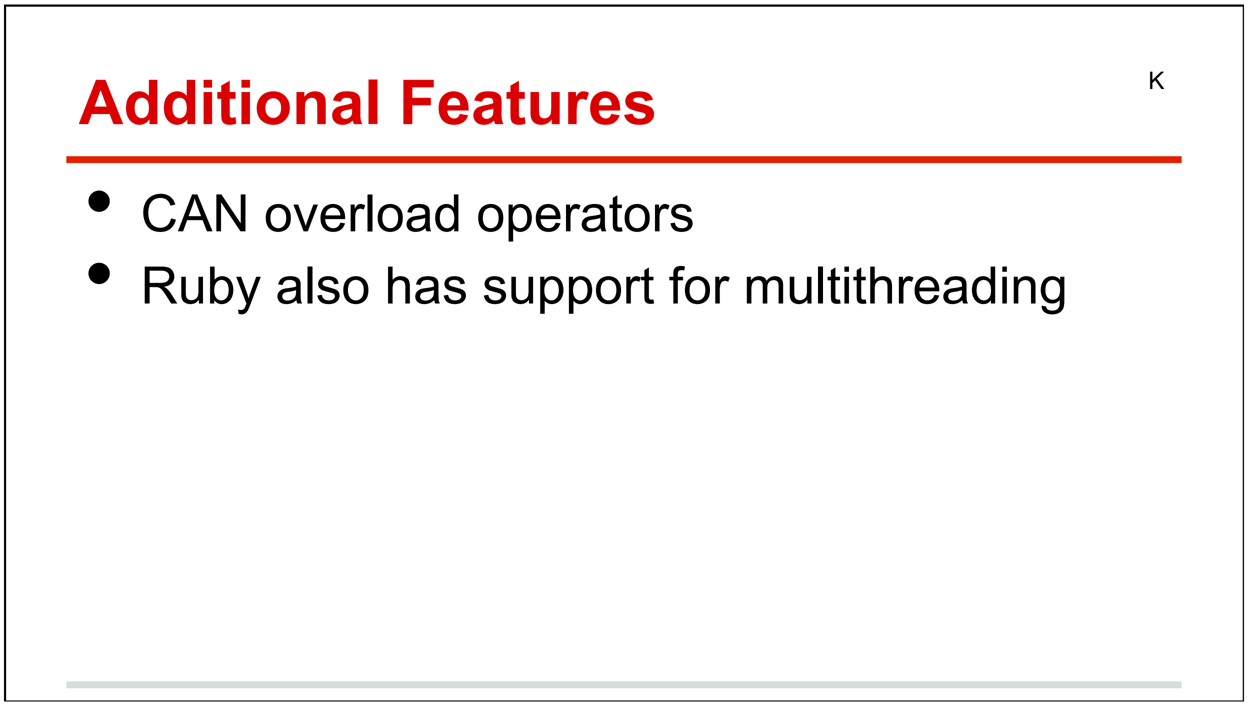
- Dynamic
- Stores primarily on heap
- Automatic memory management



Naming Conventions

K

- **Variables**
  - Variable
  - $variable
  - @variable
  - @@variable
  - variable
- **Methods**
  - ex_method?
  - ex_method!



Additional Features

K

- CAN overload operators
- Ruby also has support for multithreading

Code Example - Classes

A

```
# Class names must be capitalized.
# Technically, it's a constant.
class Fred

  # The initialize method is the constructor.  The @val is
  # an object value.
  def initialize(v)
    @val = v
  end

  # Set it and get it.
  def set(v)
    @val = v
  end

  def get
    return @val
  end
end
```

```
# Objects are created by the new method of the class
object.
a = Fred.new(10)
b = Fred.new(22)

print "A: ", a.get, " ", b.get,"\n";
b.set(34)
print "B: ", a.get, " ", b.get,"\n";

# Ruby classes are always unfinished works.
# This does not re-define Fred, it adds functionality.
class Fred
  def inc
    @val += 1
  end
end

a.inc
b.inc
print "C: ", a.get, " ", b.get,"\n";

# Objects may have methods all to themselves.
def b.dec
  @val -= 1
end
```

```
begin
  b.dec
  a.dec
rescue StandardError => msg
  print "Error: ", msg, "\n"
end

print "D: ", a.get, " ", b.get,"\n";
```

Code Example - Inheritance

A

```
# Class Barney is derived from Fred with the
# usual meaning.
class Barney < Fred
  def initialize(x)
    super(x)
    @save = x
  end

  def chk
    @save = @val
  end

  def restore
    @val = @save
  end

  def to_s
    return "(Backed-up) " + super + " [backup
value: " + @save.to_s + "]"
  end

end
```

```
# Objects are created by the new method of
the class object.
a = Fred.new(398)
b = Barney.new(112)

a.more(34)
b.more(817)

print "A: a = ", a, "\n   b = ", b, "\n";

a.more(34)
b.more(817)

print "B: a = ", a, "\n   b = ", b, "\n";

b.chk

a.more(34)
b.more(817)

print "C: a = ", a, "\n   b = ", b, "\n";

b.restore

print "D: a = ", a, "\n   b = ", b, "\n";
```

Re-Opening Classes

K

In Ruby, classes are never closed: methods can always be added to an existing class, called **Monkey-Patching**

```
# re-open Ruby's Time classclass Time
    def yesterday
      self - 86400
    endend

  today = Time.now              # => 2013-09-03 16:09:37 +0300
  yesterday = today.yesterday    # => 2013-09-02 16:09:37 +0300
```

Metaprogramming

```
COLORS = { black:   "000",
           red:     "f00",
           green:   "0f0",
           yellow:  "ff0",
           blue:    "00f",
           magenta: "f0f",
           cyan:    "0ff",
           white:   "fff" }
   class String
    COLORS.each do |color,code|
      define_method "in_#{color}" do
        "<span style=\"color: ##{code}\">#{self}</span>"
      end
    endend
```

```
"Hello, World!".in_blue
 => "<span style=\"color: #00f\">Hello, World!</span>"
```

Iteration

A

```
array = [1, 'hi', 3.14]array.each {|item| puts item }# prints:# 1# 'hi'# 3.14
    array.each_index {|index| puts "#{index}: #{array[index]}" }# prints:# 0: 1# 1: 'hi'# 2: 3.14
    # The following uses a Range(3..6).each {|num| puts num }# prints:# 3# 4# 5# 6
```

Ruby Command Prompt Example

Coding Example - Web Spider

A

```
require "rubygems"
require "anemone"

urls = File.open("urls.csv")
opts = {discard_page_bodies: true, skip_query_strings: true, depth_limit:2000, read_timeout: 10}
File.open("results.csv", "a") do |result_file| while row = urls.gets
row_ = row.strip.split(',') if row_[1].start_with?("http://")
url = row_[1] else url = "http://#{row_[1]}" end
```

```
Anemone.crawl(url, options = opts) do |
anemone|
anemone.storage = Anemone::Storage.Redis
puts "crawling #{url}"
anemone.on_every_page do |page| next if
page.body == nil if
page.body.downcase.include?("sometext")
puts "found one at #{url}"
result_file.puts "#{row_[0]},#{row_[1]}" next
end # end if end # end on_every_page end #
end crawl end # end while # we're done
puts "We're done." end # end File.open
```

Sources

http://spidr.rubyforge.org/

http://anemone.rubyforge.org/

http://www.tutorialspoint.com/ruby/ruby_object_oriented.htm

http://timetobleed.com/garbage-collection-slides-from-la-ruby-conference/

https://en.wikipedia.org/wiki/Ruby_(programming_language)#Differences_from_other_languages

http://www.jvoegele.com/software/langcomp.html

http://www.techotopia.com/index.php/Ruby_Strings_-_Creation_and_Basics

https://www.ruby-lang.org/en/documentation/success-stories/

https://stackoverflow.com/questions/8768865/regarding-to-i-method-of-ruby

http://rubylearning.com/satishtalim/ruby_exceptions.html

http://rubylearning.com/satishtalim/ruby_exceptions.html

https://www.ruby-lang.org/en/documentation/ruby-from-other-languages/

http://sandbox.mc.edu/~bennet/ruby/code/index.html