Ruby on Rails 4

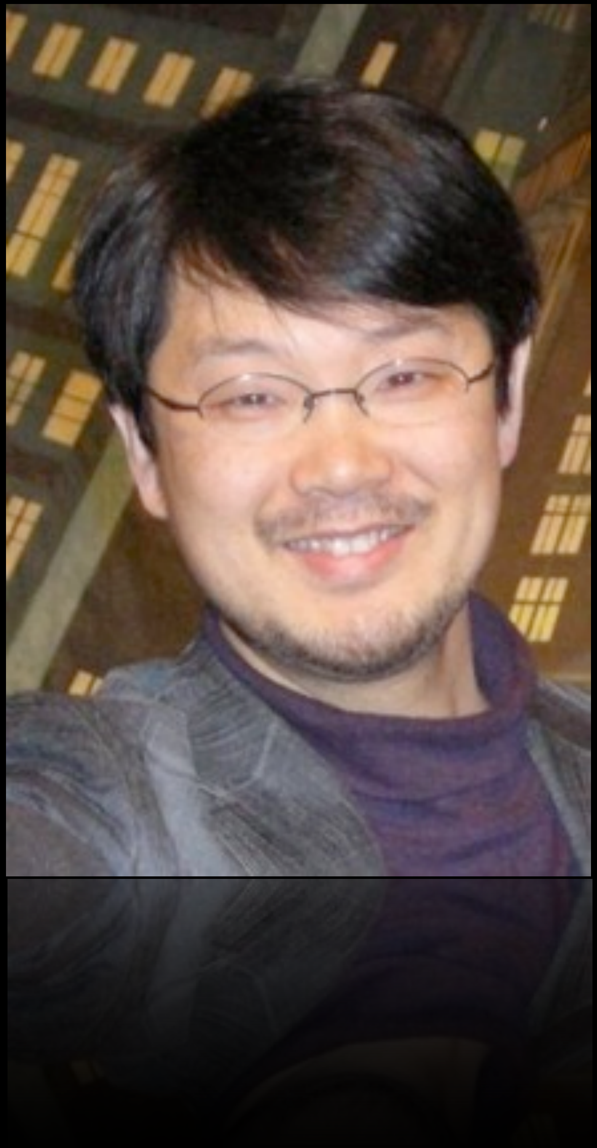
Web development workshop

Ruby on Rails

Sustainable productivity for web-application development

Rick Pannen, Consulting & Development [pannen@gmail.com]

The history of Ruby



- A scripting language like perl or python
- Developed by Yukihiro „Matz“ Matsumoto in Japan
- The first version (0.95) was released in 1995

Ruby interpreter

- MRI (Matz's Ruby Interpreter) - Reference implementation in C
- JRuby - Java implementation
- Rubinius - Ruby implementation in Ruby (uses LLVM)
- ...many more

Managing multiple Ruby interpreters

- Use rvm (Ruby Version Manager) on Linux and MacOs http://rvm.io
- Pik does the same for windows

Ruby features

- Everything is an object
- Built after the "Principle of Least Surprise"
- Encourages duck typing
- Metaprogramming, DSLs

Ruby code conventions

Constants begin with an uppercase letter.
Constant vars are uppercase by convention.

```
PI = 3.141592653589793238462643
```

Class names are CamelCase by convention

```
class MySuperClass
```

Everything else is lowercase and underscored

```
my_favorite_variable = 42
```

Ruby syntax:
As simple as possible

Commands end with semicolon or whitespace:

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

or:

```ruby
puts "Hello"; puts "World"
```

possible, but not used:

```ruby
puts "Hello";
puts "World";
```

Ruby syntax: Variable declaration

```
my_variable = "A string"
my_array = [1, 7, "Some text"]
my_hash = {:one => "Eins", :two => "Zwei"}
```

':one' is called a symbol - That is commonly used as a key in hashes.

Because of that a short form of the last statement has been introduced in Ruby 1.9

```
my_hash = {one: "Eins", two: "Zwei"}
```

Ruby:
Control structures

If - elsif - else:

```
if a == 1
  "One"
elseif a == 2
  "Two"
else
  "Three"
end
```

Ruby:
Keep it readable

Don't do this

```
if !(a == b)
```

do this:

```
unless a == b
```

on-line conditions:

```
puts "Hello" if write_hello
puts "World" unless no_world
```

Ruby:
Functions / Methods

Declaration:

```
def say_my_name
  "Rick"
end
```

The implicit return value is the last evaluated value. Just use 'return' explicitly if you need to.

Ruby: Method name conventions

Methods that return a boolean value are suffixed with a question mark:

```
"Hello".start_with? "Hell"
=> true
```

Methods that modify the object they are called on are suffixed with an exclamation mark:

```
my_string = "Hello"
my_string.reverse!
puts my_string
=> olleH
```

Ruby: Classes

Declaration:

```
class Duck
  def speak(name)
    "Quak! " + name
  end
end
```

Inheritance:

```
class Duckling < Duck
end
```

Ruby: Class instantiation

Create a new duck object:

```ruby
my_duck = Duck.new
```

Call a method on the object:

```ruby
my_duck.speak "Quack"
```

You don't need any brackets for the parameters as long as the meaning of the code is not ambiguous like this:

```ruby
my_duck.speak("Quack").downcase
```

Ruby:
Instance & class variables

Declaration

```
class Duck
  # Class variable
  @@species = "Bird"

  # Instance variable
  def initialize(name)
    @name = name
  end
end
```

Ruby:
Attribute accessors

Class and instance variables are private to the class. You need to write getters and setters to access them.

```
class Duck
  [...]
  def name
    @name
  end
end
```

Ruby:
Attribute the easy way

Writing setters and getters for all attributes would be boring so there's a simpler way:

```ruby
class Duck
 # This creates getters
  attr_reader :color, :gender
 # This creates setters
  attr_writer :weight, :size
  # This creates getters and setters
  attr_accessor :name, :location
end
```

Ruby: Modules

Modules are mixins that extend classes

```
module Named
  attr_writer :first, :last
  def full_name
    @first + " " + @last
  end
end
```

Use it in a class:

```
class User
  include Named
end
```

Ruby: Blocks

Blocks have the following syntax:

```
my_array = [1, 2, 3, 4]

my_array.each do |n|
  puts n * 2
end
```

The part in orange is a block. That is a piece of code that is passed to the method 'each' of the array 'my_array'. That method calls the code for every member of the array.

Ruby: Gems

- Gems are packaged programs and libraries for ruby
- You can install them with the "gem" commandline tool
- Type "gem install twitter" on your virtual machine to try it out

Ruby: Tutorials

http://rubymonk.com has some excellent tutorials for all skill levels.

Ruby: Questions?

- Questions
- 15 minute break
- Next up: Rails workshop

Ruby on Rails

- Model View Controller framework
- Version 1.0 was released in 2005
- 4.0, the current version, was released in June 2013

Rails: Principles

- Don’t repeat yourself
- Convention over configuration

Rails: Workshop

- Switching to terminal & editor