Introduction to Ruby

Our Goals

- History of Ruby
- Installation of Ruby
- Ruby building blocks
  - Data types
  - Variables
  - Conditionals
  - Control Structures (iteration)
  - Methods

Important Links

- The Ruby programming language
- Ruby on Github
- Ruby Docs
- Yukihiro "Matz" Matsumoto
  - Twitter
  - Github
- AirBnB Styleguide
- The Ruby Style Guide

History of Ruby

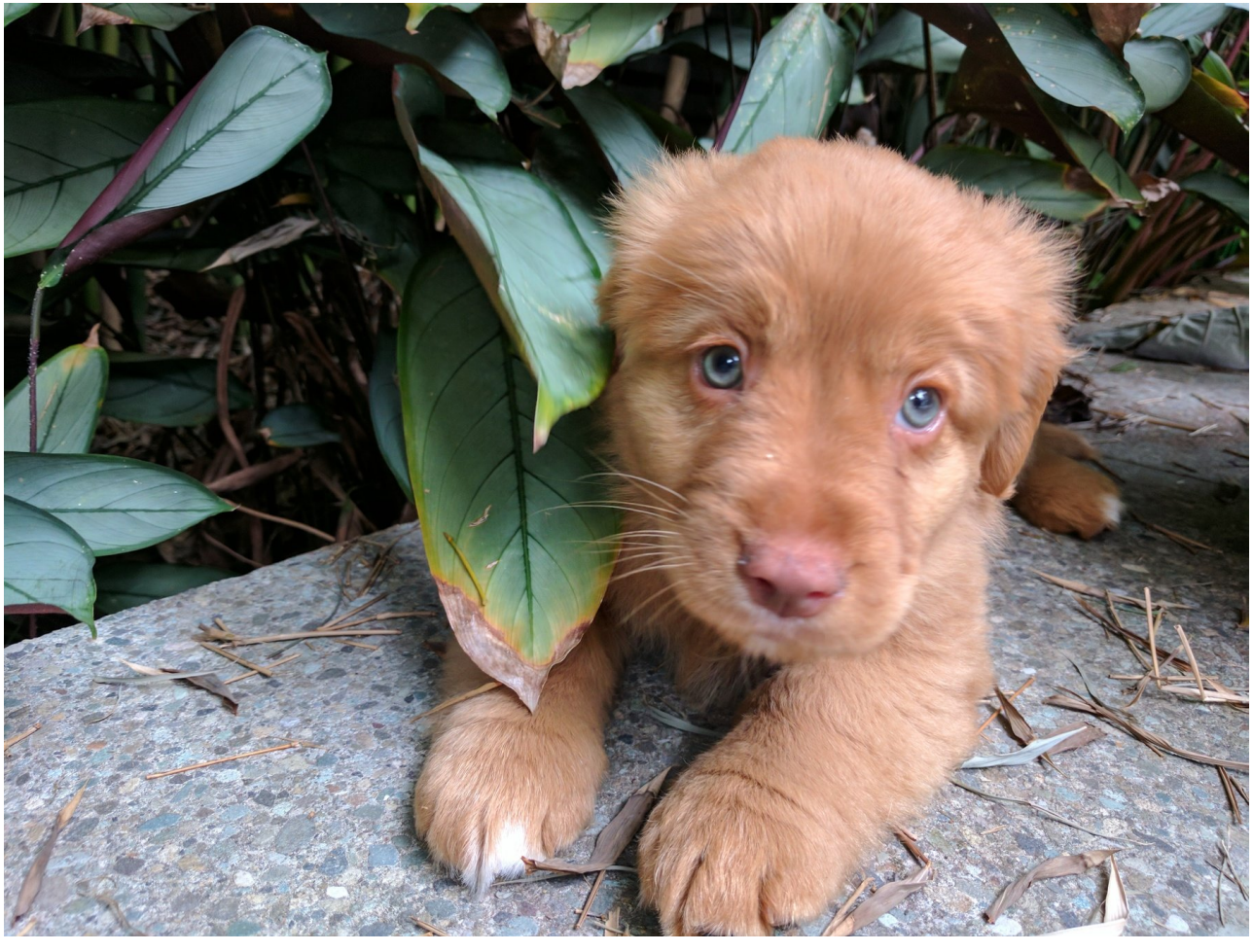
- First released in 1993
- Version 1 in 1996
- Version 1.8 in 2003
- Rails released in 2005
- Mac OS X starts having Ruby by default in 2007
- Currently at Version 2.5.0

Philosophy of Ruby

- "Making programmers happy"
- There is more than one way to do it
- There is no perfect programming language
- Principle of least astonishment

Our Official Mascot of Ruby

Our Official Mascot of Ruby



Installation of Ruby

1. Get some developer tools
2. Install RVM
3. Include RVM in your startup scripts and PATH
4. Install and use a version of Ruby
5. Install common gems

Developer Tools

```
xcode-select --install
```

Install RVM

```
curl -sSL https://get.rvm.io | bash -s stable
```

Open up your .bash_profile

```
atom ~/.bash_profile
```

Add these to the end of the file and save it

```
[[ -s "$HOME/.rvm/scripts/rvm" ]] && source "$HOME/.rvm/scripts/rvm"
export PATH="$PATH:$HOME/.rvm/bin"
```

Restart the terminal

```
rvm
rvm list known
rvm get stable --auto-dotfiles
```

Find the most recent version here

```
rvm install ruby-2.4.1
rvm --default use 2.4.1
```

Let's test that it worked

```
ruby -v
rvm -v

which ruby
```

Then install some gems

```
gem install lolcat
gem install pry
brew install fortune
brew install cowsay
brew install ponysay
brew install cmatrix
```

Some common commands

- `ruby -v`
- `which ruby`
- `ruby hello_world.rb`
- `irb`
- `pry`
- `<CTRL> + D`

Data Types

- Strings
- Numbers
- Arrays
- Hashes (like objects)
- Methods (like functions)
- Symbols

Strings

```
# Double and single quotes will both work,
# but there are a few differences

'Hello World'
"Hello World"

# Double quotes have interpolation!

'2 + 2 = #{ 2 + 2 }'
"2 + 2 = #{ 2 + 2 }"

# You can see all the methods!

"Hello World".methods
```

Arithmetic

```
10 + 4
10 - 6
10 * 12
10 / 12

10 < 12
12 > 10
10 >= 10
12 <= 12

10 == 10 # Use double equals in Ruby!
10 === 10

10 != 9
```

Numbers

```
1.0
2.1512

1241
125125129

1294810294801284012840812908
2512159412125699832859328

# Behind the scenes...
# Complex, Rational, Bignum
# Float, Fixnum, Integer, BigDecimal
```

Variables

```ruby
this_is_ruby = true
this_is_a_string = "Yes, it is"
this_is_a_number = 1241
this_is_a_number += 1
this_is_a_number -= 1

empty_array = []
empty_hash = {}

name = "Gilberto"
drink = "Whiskey"

"My name is #{ name } and I drink #{ drink }"

# variable names can only start with a-z or _
12monkeys = 'film'  # => SyntaxError
```

Getting user input

```
puts "What is your first name? "

first_name = gets
first_name = gets.chomp      # better

puts "Your first name is #{ first_name }"

puts "What is your last name? "

last_name = gets.chomp

puts "Your surname is #{ last_name }"

puts "Your full name is #{ first_name } #{ last_name }"
```

Conditionals - IF

```
if 42 > 13
    p "42 is a bigger number"
end

name = "Groucho"
if name == "Harpo"
    # Do something
elsif name == "Chico"
    # Do something else
else
    # Do something else
end

p "42 is bigger" if 42 > 13
```

Conditionals - UNLESS

```
x = 1
unless x > 2
    puts "x is less than 2"
else
    puts "x is greater than 2"
end

code_to_perform unless conditional
```

Conditionals - CASE

```ruby
# You can use a case statement to rewrite a chain of
# if-elsif-elsif statements into a more readable form
hour = 15

case hour
when 12 then  puts "Lunchtime"
when 15 then  puts "Naptime"
else          puts "Whatever!"
end

# You can also use ranges
case hour
when 8..12
  puts "Morning"
when 13..17
  puts "Afternoon"
end
```

Conditionals - CASE

```ruby
# You can use a case statement to perform any kind
# of conditional test you want, but in that case (lol)
# you must omit the variable from the 'case' line
# and use the full expression in each 'when' line

hour = 15

case
when hour < 12
    puts "Good Morning"
when hour > 12 && hour < 17
    puts "Good Afternoon"
else
    puts "Good Evening"
end
```

Logical Operators

```
true && true
true and true

true || false
true or false

!true

# TRUTHINESS IN RUBY:
# ONLY false AND nil ARE FALSEY
# EVERYTHING ELSE IS TRUTHY... EVEN 0 !!!
```

Have a crack at these exercises

Loops - WHILE

```ruby
while conditional
    # Statements to execute
end

while true
    puts "This is a great idea"
end

i = 0
while i < 5
    puts "I: #{ i }"
    i += 1
end
```

Loops - UNTIL

```
until conditional
    # Statements to execute
end

i = 0
until i == 5
    puts "I: #{ i }"
    i += 1
end
```

Loops - ITERATORS

```ruby
5.times do
    puts "Wow"
end

5.times do |i|
    puts "I: #{i}"
end

5.downto(0) do |i|
    puts "I: #{ i }"
end

5.upto(10) do |i|
    puts "I: #{ i }"
end
```

Loops - FOR

```
# DON'T USE THEM!

for i in 0..5
   puts "I: #{ i }"
end
```

Generating random numbers

```
# Generates a number between 0 and 1
Random.rand

# Generates a random number up to 10
# (including zero, but not 10 itself)
Random.rand(10)

# Generates a number between 5 and 10 (also includes them)
Random.rand(5..10)

# Does not include 10
Random.rand(5...10)
```

Have a crack at these exercises

Methods

```
def hello
    puts "Hello World"
end

hello
hello()

# GOTCHA: No space allowed between name and the ()
hello ()
# => "ArgumentError: wrong number of arguments
#     (given 1, expected 0)"


def hello( name )
    puts "Hello #{ name }"
end

hello "Roget"
hello( "Roget" )
```

Implicit Return

```ruby
def add( first, second )
  result = first + second
  return result
end

def add( first, second )
  result = first + second
  # no need for return keyword: value of
  # method's last line is implicitly returned
  result
end

def add( first, second )
  puts "adding #{first} + #{second}"
  first + second   # no need for a temporary variable
end
```

Here is your homework