♦

# Ruby

| | |
|---|---|
| Category | Programming |
| Created | @Jan 21, 2021 8:45 PM |
| Files | |
| Official Documentation | https://www.ruby-lang.org/en/documentation/ |
| Reminder | |
| Status | Open |
| Taken From | Youtube |
| URL | https://www.youtube.com/watch?v=8wZ2ZD--VTk&t=135s |
| Updated | @Jan 21, 2021 10:56 PM |

## Ruby Overview

Ruby is a general purpose, dynamically typed and reflective, object oriented programming language that was created in the 1990s by Yukihiro Matsumoto.

The original intent when creating ruby was to create a true Object Oriented scripting language. Which at the time, in Matsumoto's optinion, didn't exist. Ruby was designed to be simple at it's core, but with an object system that was fully integreated into the language, and not tacked on as a glorified addon.

At the core of ruby is the desire for programmer productivity and fun. Ruby's core focus is on the programmer not the machine. Therefore ruby attempts to remove as much confusion as possible from the programming process by utilizing a simple, readable syntax and deeply integrated object orientation.

### How Ruby Runs

Generally all ruby code is run using an interpreter, although with some implementations of ruby it is possible to compile the code and run it on a virtual machine.

The most popular ruby interpreter is called MRI which stands for Matz's Ruby Interpreter. Unfortunatly there is no official Ruby Language Reference, so generally the Matz interpreter is seen as the language standard.

Ruby utilizes a garbage collector and it's syntax is very minimal and simple.

One reason Ruby has become so popular is due to the Ruby on Rails web application framework. Rails is extremely easy to use which is why it's the framework of choice for tons of large companies like Github, Twitch and Hulu.

### Choosing an IDE

Many developers choose to write Ruby using a basic text editor, but there are also more specilized integrated development enviornments, some of the most popular include RubyMine, Aptana RadRails and Vim.

## Code

### Printing

```
puts "Hello"
print "World"
puts "!"
```

## Variables and Data Types

```
=begin
 Names are case-sensitive and may begin with:
     letters, _
 After, may include
     letters, numbers, _
 Convention says
     Start with a lowercase word, then additional words are lowercase separated
     by an underscore
     ex. my_first_variable
=end
name = "Mike"     # Strings
age = 30          # Integer
gpa = 3.5         # Decimal
is_tall = true    # Boolean -> True/False

name = "John"

puts "Your name is #{name}"
puts "Your name is " + name
```

## Casting and Converting

```
puts  3.14.to_i
puts  3.to_f
puts "3.0".to_s

puts 100 + "50".to_i
puts  100 + "50.99".to_f
```

## Strings

```
greeting = "Hello"
#indexes:   01234

puts greeting.length
puts greeting[0]
puts greeting.include? "llo"
puts greeting.include? "z"
puts greeting[1,3]
```

## Numbers

```
puts  2 * 3          # Basic Arithmetic: +, -, /, *
puts  2**3           # Exponent
puts  10 % 3         # Modulus Op. : returns remainder of 10/3
puts  1 + 2 * 3      # order of operations
puts 10 / 3.0        # int's and doubles


num = 10
num += 100           # +=, -=, /=, *=
puts num

num = -36.8
puts  num.abs()
puts  num.round()

# Math class has useful math methods
puts Math.sqrt(144)
puts Math.log(0)
```

## User Input

```
puts "Enter your name: "
name = gets                              #.chomp
puts "Hello #{name}, how are you"

puts "Enter first num: "
num1 = gets.chomp
puts "enter second num: "
num2 = gets.chomp
puts num1.to_f + num2.to_f
```

## Arrays

```
lucky_numbers = [4, 8, "fifteen", 16, 23, 42.0]
#       indexes  0  1       2      3   4   5

lucky_numbers[0] = 90
puts lucky_numbers[0]
puts lucky_numbers[1]
puts lucky_numbers[-1]

puts "\n\n"
puts lucky_numbers[2,3]
puts "\n\n"
puts lucky_numbers[2..4]
puts "\n\n"

puts lucky_numbers.length
```

## 2 Dimensional Arrays

```
number_grid = [[],[]]
# numberGrid = [ [1, 2], [3, 4] ]
number_grid[0][0] = 99

puts number_grid[0][0]
puts number_grid[0][1]
```

## Array Methods

```
friends = []
friends.push("Oscar")
friends.push("Angela")
friends.push("Kevin")

# friends.pop
puts  friends
puts "\n"

puts friends.reverse
puts "\n"

puts friends.sort
puts "\n"

puts  friends.include? "Oscar"
```

## Methods

```
def add_numbers(num1, num2=99)return num1 + num2
end

sum = add_numbers(4, 3)
puts sum
```

## If Statements

```
is_student = false
is_smart = false

if is_student and is_smart
  puts "You are a student"
elsif is_student and !is_smart
  puts "You are not a smart student"
else
  puts "You are not a student and not smart"
end

# >, <, >=, <=, !=, ==, String.equals()if 1 > 3
  puts "number comparison was true"
end

if "a" > "b"
     puts "string comparison was true"
end
```

## Switch Statements

```
my_grade = "A"
case my_grade
     when "A"
    puts "You Pass"
     when "F"
      puts "You fail"
     else
      puts "Invalid grade"
end
```

## Dictionaries

```
test_grades = {
    "Andy" => "B+",
    :Stanley => "C",
    "Ryan" => "A",
    3 => 95.2
}

test_grades["Andy"] = "B-"
puts test_grades["Andy"]
puts test_grades[:Stanley]
puts test_grades[3]
```

## While Loops

```
index = 1
while index <= 5
  puts index
  index += 1
end
```

## For Loops

```
for index in 0..5
    puts index
end

5.times do |index|
    puts index
end

lucky_nums = [4, 8, 15, 16, 23, 42]
for lucky_num in lucky_nums
    puts lucky_num
end

lucky_nums.each do |lucky_num|
     puts lucky_num
end
```

## Exception Catching

```
begin
     # puts bad_variable
     num = 10/0
rescue ZeroDivisionError
     puts "Error"
rescue
     puts "All other errors"
end

raise "Made Up Exception"
```

## Classes and Objects

```
class Book
  attr_accessor :title, :author

     def readBook()
```

```
          puts "Reading #{self.title} by #{self.author}"
     end
end

book1 = Book.new()
book1.title = "Harry Potter"
book1.author = "JK Rowling"

book1.readBook()
puts book1.title
```

## Constructors

```
class Book
    attr_accessor :title, :author
        def initialize(title, author)
            @title = title
            @author = author
        end

        def readBook()
            puts "Reading #{self.title} by #{@author}"
        end
end

book1 = Book.new("Harry Potter", "JK Rowling")
    # book1.title = "Half-Blood Prince"

puts book1.title
```

## Getters and Setters

```
class Book
  attr_accessor :title, :author
        def initialize(title, author)
             self.title = title
            @author = author
        end

        def title=(title)
            puts "Set"
            @title = title
        end
        def title
            puts "Get"
            return @title
        end
end

book1 = Book.new("Harry Potter", "JK Rowling")

puts book1.title
```

## Inheritance

```
class Chef
      attr_accessor :name, :age
     def initialize(name, age)
          @name = name
          @age = age
     end

     def make_chicken()
          puts "The chef makes chicken"
     end

     def make_salad()
          puts "The chef makes salad"
     end

     def make_special_dish()
          puts "The chef makes a special dish"
     end
end

class ItalianChef < Chef

     attr_accessor :country_of_origin
     def initialize(name, age, country_of_origin)
          @country_of_origin = country_of_origin
```

```ruby
          super(name, age)
     end

     def make_pasta()
          puts "The chef makes pasta"
     end

     def make_special_dish()
          puts "The chef makes chicken parm"
     end
end

my_chef = Chef.new()
my_chef.make_chicken()

my_italian_chef = ItalianChef.new()
my_italian_chef.make_chicken()
```