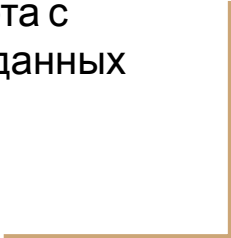


Начало работы с Ruby

История языка, работа с
числами, ввод и вывод данных



История языка Ruby

- Ruby был задуман в 1993-м году японцем Юкихиро Мацумото
- Первая доступная версия появилась в 1995-м году
- Скачок популярности произошел в 2003-м году с выходом Ruby on Rails 2.0
- В 2009-м году вышла версия 1.9.1, в которой была увеличена скорость работы интерпретатора
- В данный момент довольно часто используется при разработке (<https://www.tiobe.com/tiobe-index/>)

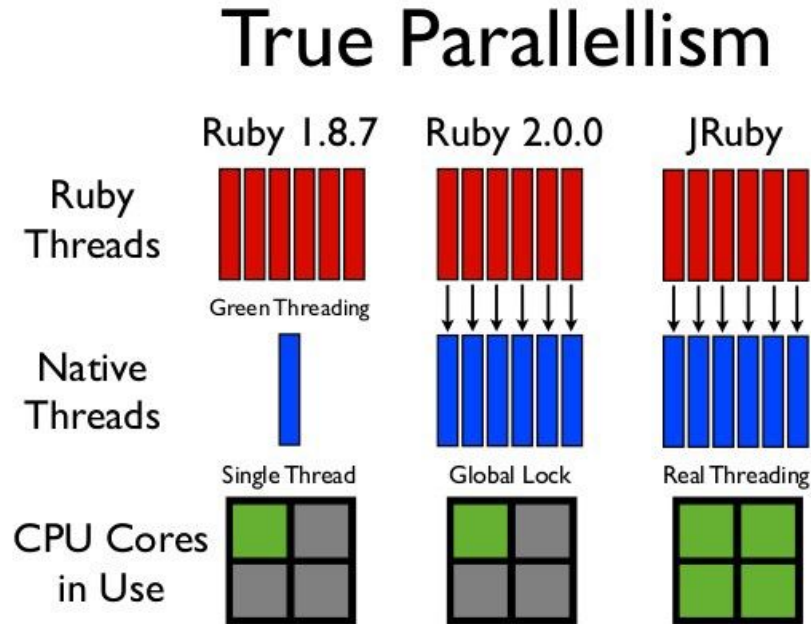


Философия Ruby

- Ruby - полностью объектно-ориентированный язык: все примитивные типы - объекты; все функции - методы
- По задумке создателя, язык должен быть более мощным, чем Perl, и более объектно-ориентированный, чем Python
- Язык должен быть удобным для человека

Недостатки?

- Низкая производительность
- Запутанное, непривычное ООП
- Нет статической или явной типизации
- синхронизация потоков с помощью GIL



Работа с переменными

```
1  simple_var = 1
2  simple_var = 'It is string now'
3
4  _ = 'Unused variable'
5  # 3_ = 3 # Wrong
6
7  puts 'Values', simple_var, _
8
9  $global_var = 123
10
11 print "Global val: ", $global_var, "\n"
12
13 @instance_variable = true
14
15 puts 'Instance variables:', @instance_variable, @unknown_variable
16
17 class A
18   @@class_variable = 0
19
20   def class_var
21     @@class_variable
22   end
23 end
24
25 puts 'Class variable:', A.new.class_var
26
27 CONST_VAL = 'Const'
28 CONST_VAL = 'Changed Const'
29
30 print 'Const: ', CONST_VAL, "\n"
```

Работа с методами

```
1  def new_method
2    3 + 2
3  end
4
5  def method_with_params(a, b = 0)
6    a + b
7  end
8
9  def method_with_named_params(a:, b: 0)
10   a + b
11 end
12
13 def method_with_many_params(param1, param2, optional1:, optional2:)
14   return optional1 + optional2 if optional1 == 1
15
16   param1 + param2
17 end
18
19 def method_with_array(*arr)
20   arr[0]
21 end
22
23 def method_with_hash(**hash)
24   hash[:a]
25 end
26
27 def difficult_method(a, b = 0, *arr, **hash)
28   puts "a = #{a}, b = #{b}, arr = #{arr}, hash = #{hash}"
29 end
30
```

Ввод данных

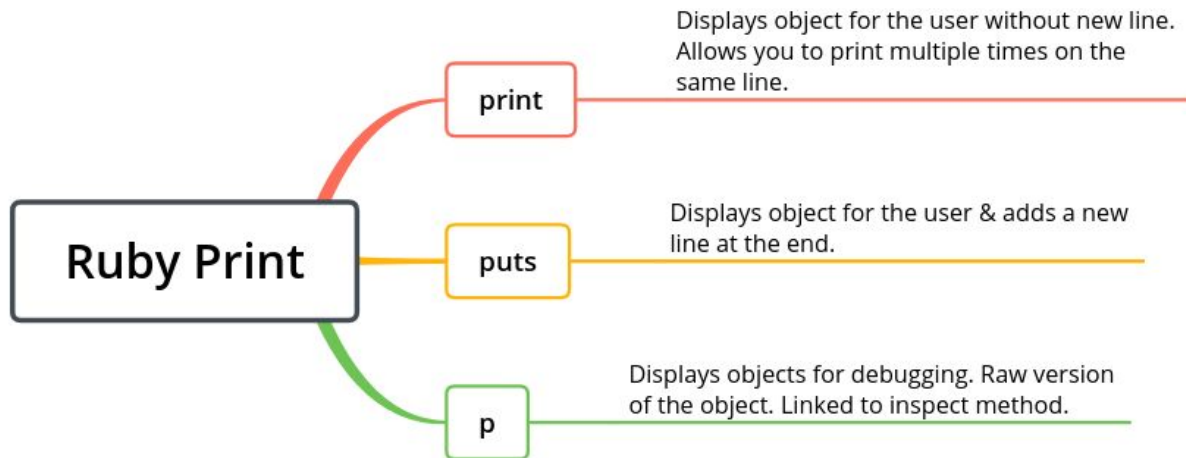
Ожидание ввода пользователя:

```
9   get_value = gets
10  puts get_value
11
12  puts gets.class
```

Чтение аргументов, которые были переданы при запуске:

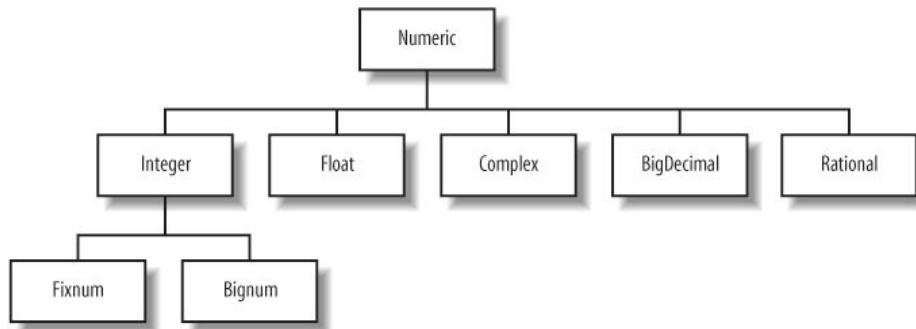
```
9  puts ARGV
```

Вывод данных



```
1 puts 123, "string \n new line", [1, 2, nil], nil
2
3 print 123, "string \n new line", [1, 2, nil], nil
4
5 p 123, "string \n new line", [1, 2, nil], nil
6
7 pp 123, "string \n new line", [1, 2, nil], nil
8
```


Работа с числами



```
1  123
2  puts 123.class
3
4  puts 1_234_567.class
5
6  puts 0123.class
7  puts 0b1111.class
8
9  puts 0b1111, 0b1111.to_s(10), 15.to_s(2)
10
11 -500
12
13 puts 12345678901234567890123456789012345678901234567890.class
14
15 puts 123.4.class
16 puts 1e6.class, 1e6, 1e-6
```

Работа с массивами чисел

```
--  
18 puts [1, 2, 'string', nil, false]  
19  
20 [1, 2, 'string', nil, false].each { |elem| puts elem }  
21 puts [1, 2, 3].map { |elem| elem**2 }  
22  
23 arr = [1, 2, 3]  
24 p arr.shift, arr  
25 p arr.pop, arr  
26
```

Конец! Спасибо!