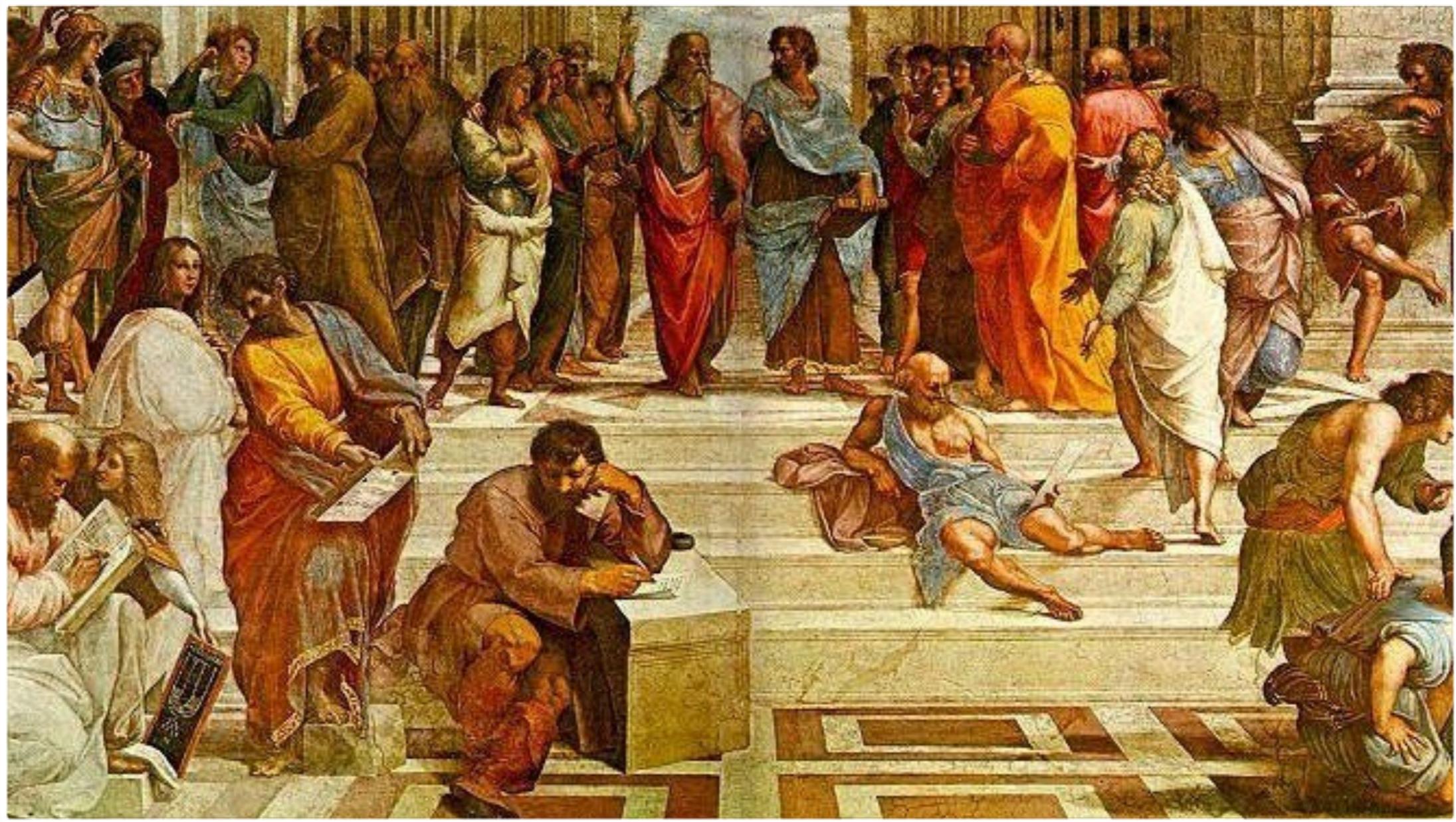


РУЗУ



Интерпретируемый, динамичный, человеколюбивый



ФИЛОСОФИЯ RUBY

твой путь — твои решения

- Принцип наименьшей неожиданности
- Минимальное время не исполнения программы, но разработки
- Множество возможных путей решения задачи
- Просто, но не примитивно
- Программы должны писаться так, чтобы их понимал человек, и лишь иногда - машина

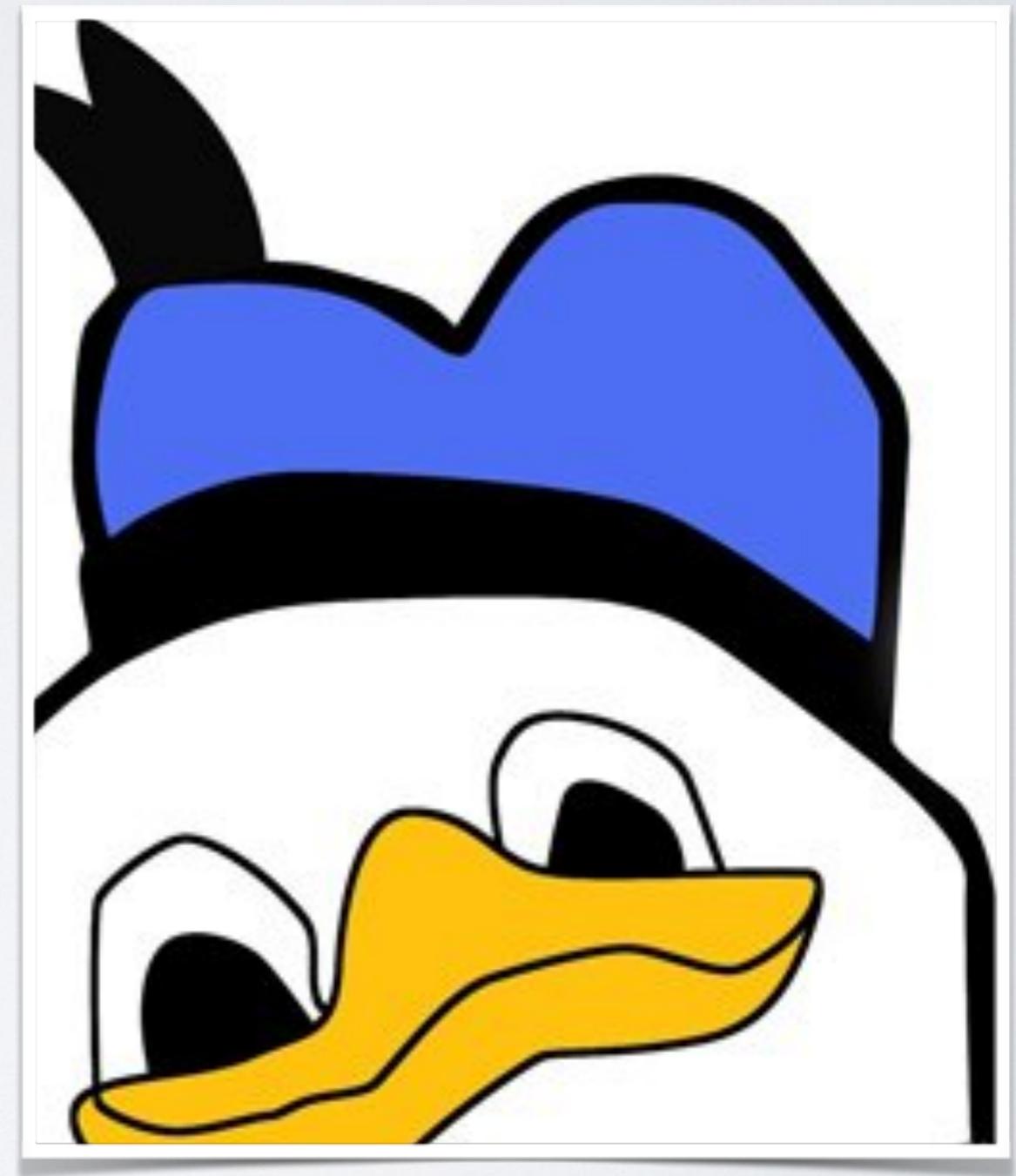


ОБЪЕКТНАЯ МОДЕЛЬ РУБИ

вместо Адама — BasicObject

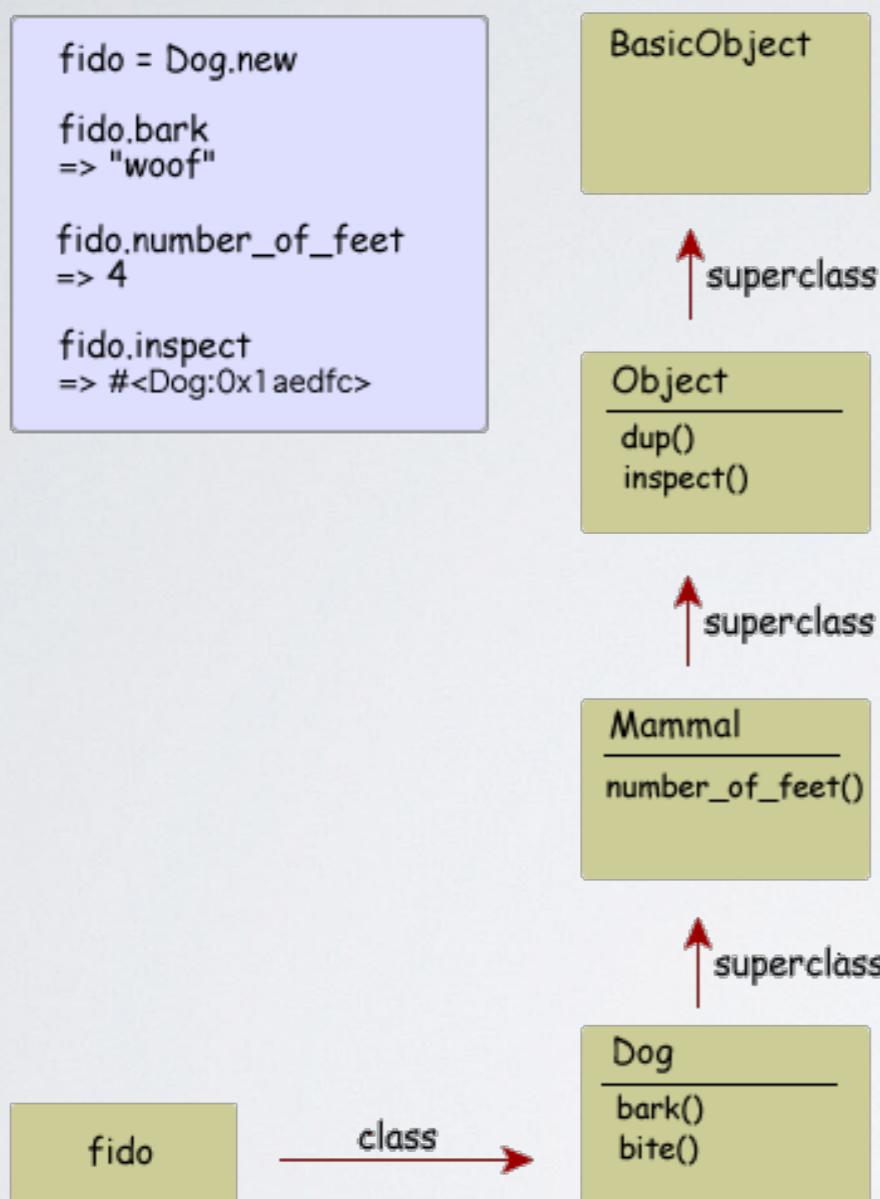
ВСЕ ОЧЕНЬ ХОРОШО!

- Нет примитивных типов
- Каждый объект - экземпляр класса
- Нет абстрактных классов
- Нет множественного наследования
- Mixins предпочтительней наследования
- Утиная типизация

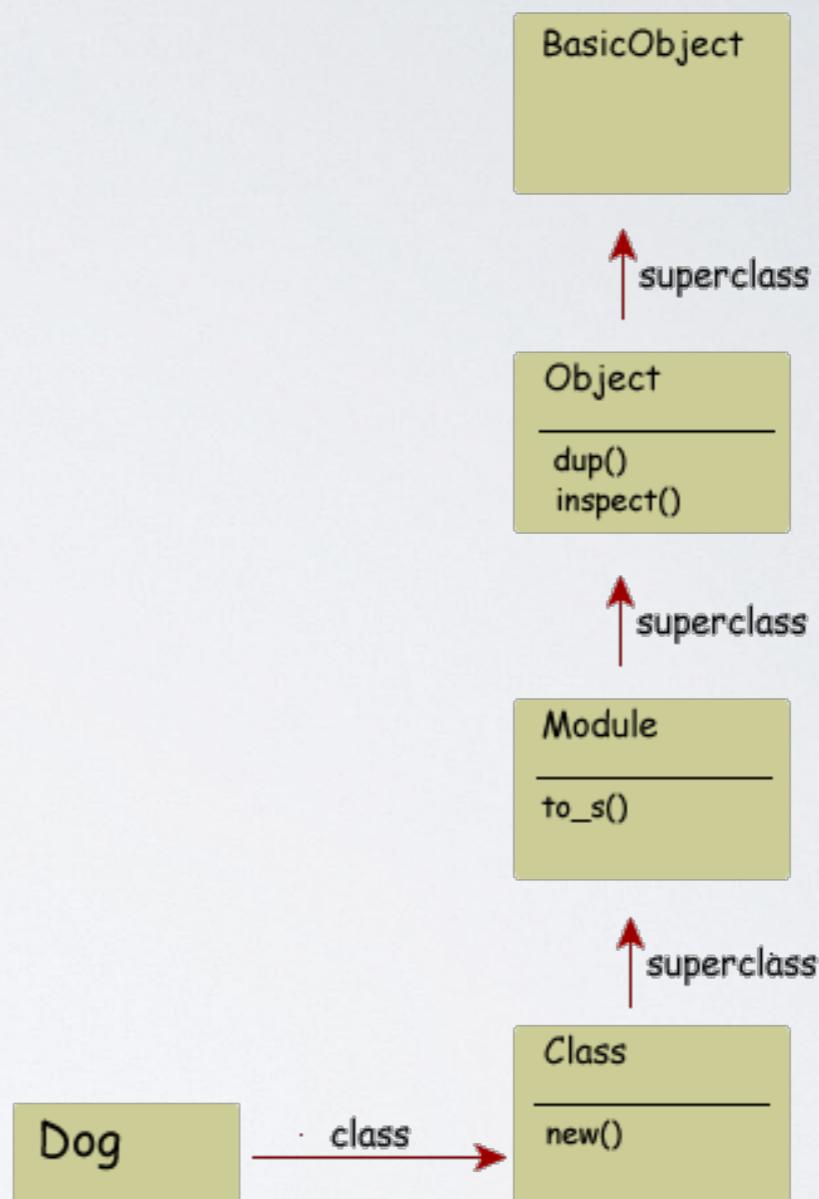


**Класс - это конструктор
объектов (экземпляров
класса, *instances of class*).**

```
fido = Dog.new  
  
fido.bark  
=> "woof"  
  
fido.number_of_feet  
=> 4  
  
fido.inspect  
=> #<Dog:0x1aedfc>
```



**Но и сам класс - это
объект в цепочке
этих конструкторов!**



Классы, выступая объектами, становятся производными
от анонимных синглтон-классов (singleton class).

Модуль — это именованная группа, которая содержит методы, которые можно подмешивать к

Объекту

include:

добавляет методы модуля объекту.

```
1 module MyModule
2   def goal
3     'Служить экземпляру класса!'
4   end
5 end
6
7 class MyClass
8   include MyModule
9 end
10
11 puts MyClass.new.goal
12 # => 'Служить экземпляру класса!'
13 puts MyClass.goal
14 # => [NoMethodError] undefined method
15 # `goal' for MyClass:Class
```

Классу

extend:

вызывает **include** для синглтон-класса объекта.

```
1 module MyModule
2   def goal
3     'Служить классу!'
4   end
5 end
6
7 class MyClass
8   extend MyModule
9 end
10
11 puts MyClass.goal
12 # => 'Служить классу!'
13 puts MyClass.new.goal
14 # => [NoMethodError] undefined method
15 # `goal' for #<MyClass:0x007fa49a081768>
```

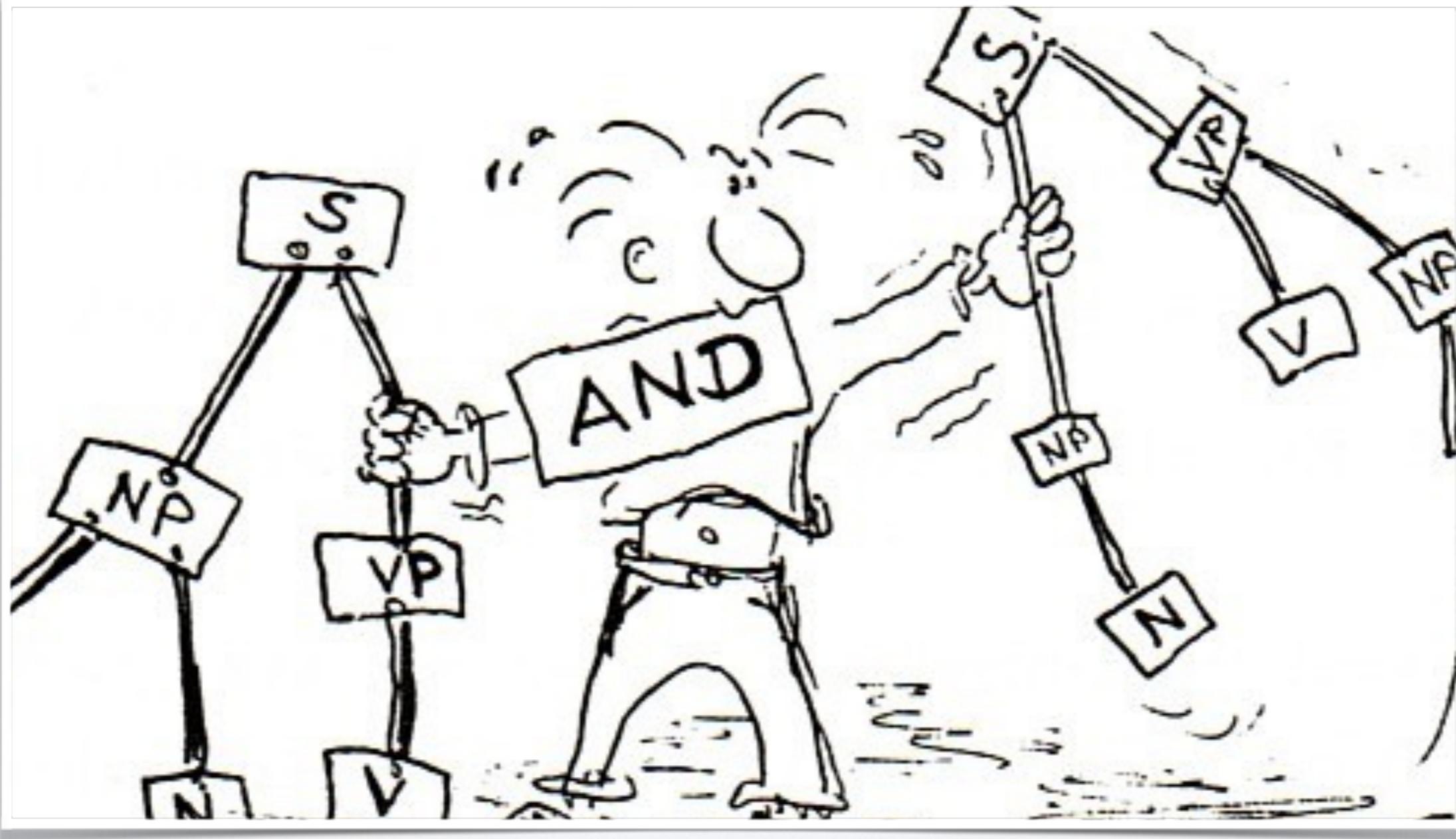
МОДУЛИ (КОНТЕЙНЕРЫ МЕТОДОВ)

```
1 module Module1
2   def method
3     puts 'Module1#method'
4   end
5
6   def ancestor_method
7     super
8   end
9 end
10
11 module Module2
12   def method
13     puts 'Module2#method'
14   end
15
16   def ancestor_method
17     super
18   end
19 end
20
21 class Parent
22   def method
23     puts 'Parent#method'
24   end
25   alias :ancestor_method :method
26 end
```

```
28 class Child < Parent
29   include Module1
30   include Module2
31
32   def method
33     puts 'Child#method'
34     puts super
35   end
36
37   def ancestor_method
38     super
39   end
40 end
41
42 Child.new.method
43 # => выведет "Child#method",
44 # а потом "Module2#method"
45 Child.new.ancestor_method
46 # => Parent#method
47 Child.ancestors
48 # => [Child, Module2, Module1, Parent,
49 #       Object, Kernel, BasicObject]
50
```

ПОЛЬЗОВАТЕЛЬСКИЕ КЛАССЫ

```
1 # $x глобальная переменная
2 # @x - инстанс-переменная экземпляра класса
3 # @@x - переменная класса
4 # у - локальная переменная с ограниченной областью видимости
5
6 class MyClass
7   attr_reader :x
8
9   @@x = 'class variable'
10  def initialize
11    y = 'local variable'
12    @x = 'instanse object variable'
13  end
14
15  def get_class_variable
16    @@x
17  end
18 end
19
20 object = MyClass.new
21 object.x #=> 'instanse object variable'
22 object.get_class_variable #=> 'class variable'
23 y # => [NameError] undefined local variable or method `y' for main:Object
```



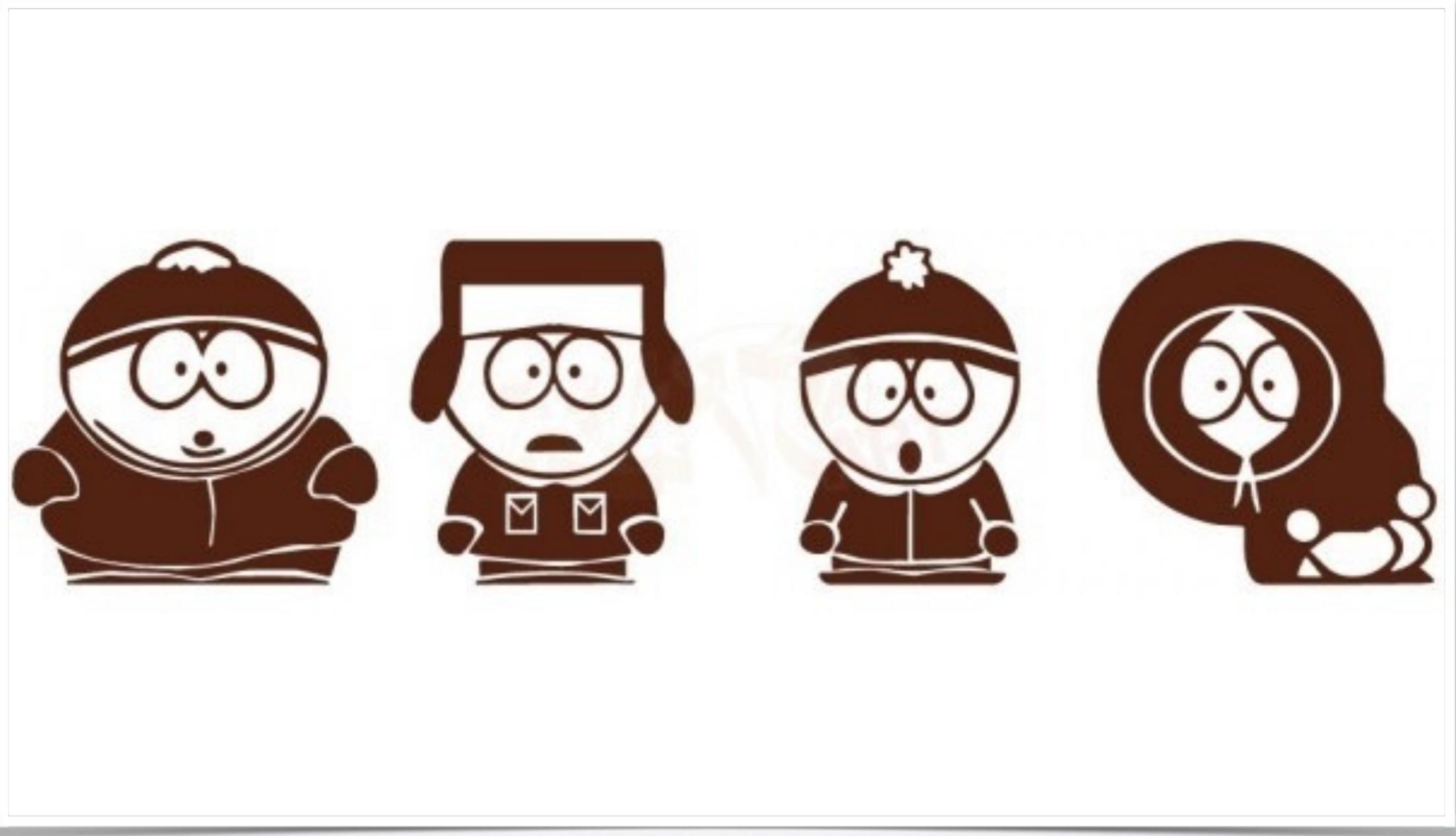
ОСОБЕННОСТИ СИНТАКСИСА

минимум управляющих конструкций

БАЗОВЫЕ ОПЕРАЦИИ

```
1 # присвоение
2 x = 1
3 x = 'string'
4 x = [1, 2]
5
6 # сравнение
7 1 == 2 #=> false
8
9 # продвинутое сравнение
10 1 <=> 2 #=> -1
11
12 # условия
13 if 1 == 1
14   true
15 else
16   false
17 end #=> true
18
```

```
18
19 'correct condition' if 1 == 1 #=> 'correct condition'
20
21 0 == 1 ? true : false          #=> false
22
23 # все операции в ruby – методы
24 1.==(2) #=> false
25 1.<=>(2) #=> -1
26
27 # методы конвертации
28 1.to_s  #=> '1'
29 '1'.to_i #=> 1
30
31 # переменная – это указатель на объект
32 s1 = 'Hello World'
33 s2 = s1
34 p s1.object_id # => 70144764317700
35 p s2.object_id # => 70144764317700
36
37
```



БАЗОВЫЕ КЛАССЫ

String, Symbol, Numeric, Hash, Array

STRING

```
1 # инициация
2 'hello world!'.class #=> String
3 "hello world!"
4 %{hello world!}
5 String.new('test').class #=> String
6
7 # экранирование
8 'it\\'s test example' #=> it's test example
9 "symbol \\\"A\\\" "#=> symbol "A"
10
11 # конкатенация
12 s1 = 'my '
13 s2 = 'test'
14
15 s1 + s2 #=> my test
16 s1 # => my
17
18 s1 << s2 #=> my test
19 s1 # => my test
20
21 # подстановка в строку
22 my_age = 29
23 'my age: ' + my_age #=> [TypeError] no implicit conversion of Fixnum into String
24 "my age #{my_age}" #=> my age: 29
```

SYMBOL

```
1 :symbol.class          #=> Symbol
2
3 # один и тот же символ всегда имеет один и тот же id
4 :symbol.object_id     #=> 384328
5 :symbol.object_id     #=> 384328
6 'string'.object_id   #=> 70273663768340
7 'string'.object_id   #=> 70273663768280
8
9 # может создаваться на базе строки
10 s = 'thing'
11 :"#{s}".object_id    #=> 405448
12 :thing.object_id     #=> 405448
13
14 # конвертация
15 :thing.to_s           #=> 'thing'
16 'thing'.to_sym        #=> :thing
17
```

NUMERIC

```
1  (2**30-1).class #=> Fixnum
2  (2**30).class #=> Bignum
3  1.1.class #=> Float
4
5
6  4/3 #=> 1
7  (4/3).class #=> Fixnum
8
9  4.0/3 #=> 1.333333333333333
10 (4/3.0).class #=> Float
11
12 rand(100) #=> random Fixnum value from 0 to 99
13 rand #=> random Float value (like 0.487613333333333)
14
```

ARRAY

```
1 # массив это упорядоченная по индексу коллекция объектов
2 array = Array.new #=> []
3 Array.new(3)      #=> [nil, nil, nil]
4 Array.new(3, true) #=> [true, true, true]
5 [true, true, true] #=> [true, true, true]
6
7 # может быть смешанного типа
8 [1, 'test', false, nil] #=> [1, 'test', false, nil]
9
10 # многомерные массивы
11 empty_table = Array.new(3) { Array.new(3) }
12 #=> [[nil, nil, nil], [nil, nil, nil], [nil, nil, nil]]
13
```

HASH

```
1 # хэш это неупорядоченная группа пар ключ-значение
2 array = Hash.new          #=> {}
3 { :k1 => 'a', :k2 => 'b' } #=> :k1=>"a", :k2=>"b"
4 { k1: 'a', k2: 'b' }      #=> :k1=>"a", :k2=>"b"
5
6 # определяем хэш со значением по умолчанию
7 h = Hash.new(5)
8 h[:any_key]               #=> 5
9
10 # ключами и значениями могут быть объекты любого класса
11 h = { :key => 'value' }
12 { h => :test, 1 => 2 }    #=> { { :key=>"value" }=>:test, 1=>2 }
13
```

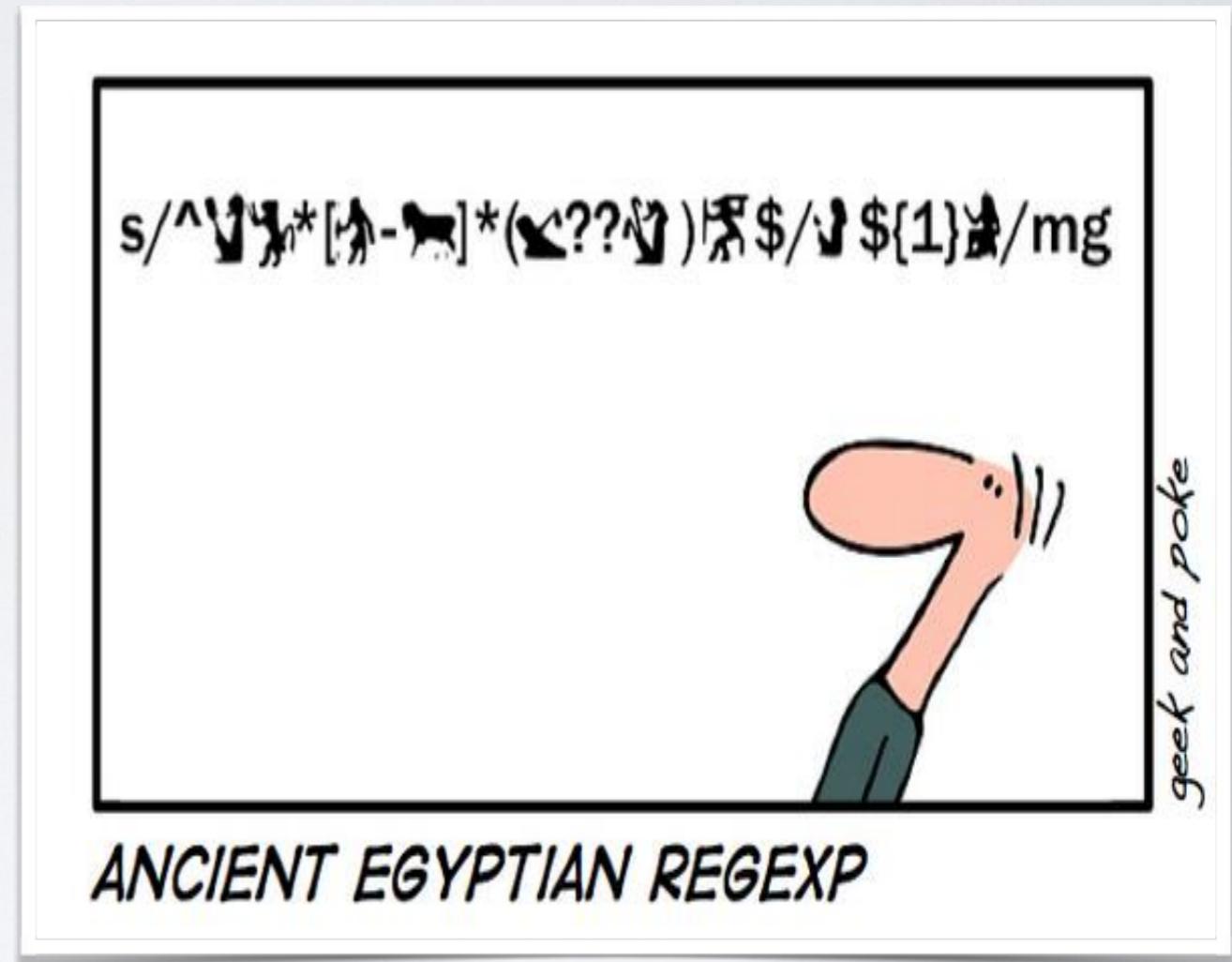
ENUMERATORS

```
1 3.times do
2   puts "Hello"
3 end
4
5 (0..3).each do |i|
6   puts i*i
7 end
8
9 %w(январь февраль март).each do |m|
10  puts m
11 end
12
13 0.upto(3) do |k|
14  puts k
15 end
16
17 4.downto(1) do |t|
18  puts t**3
19 end
```

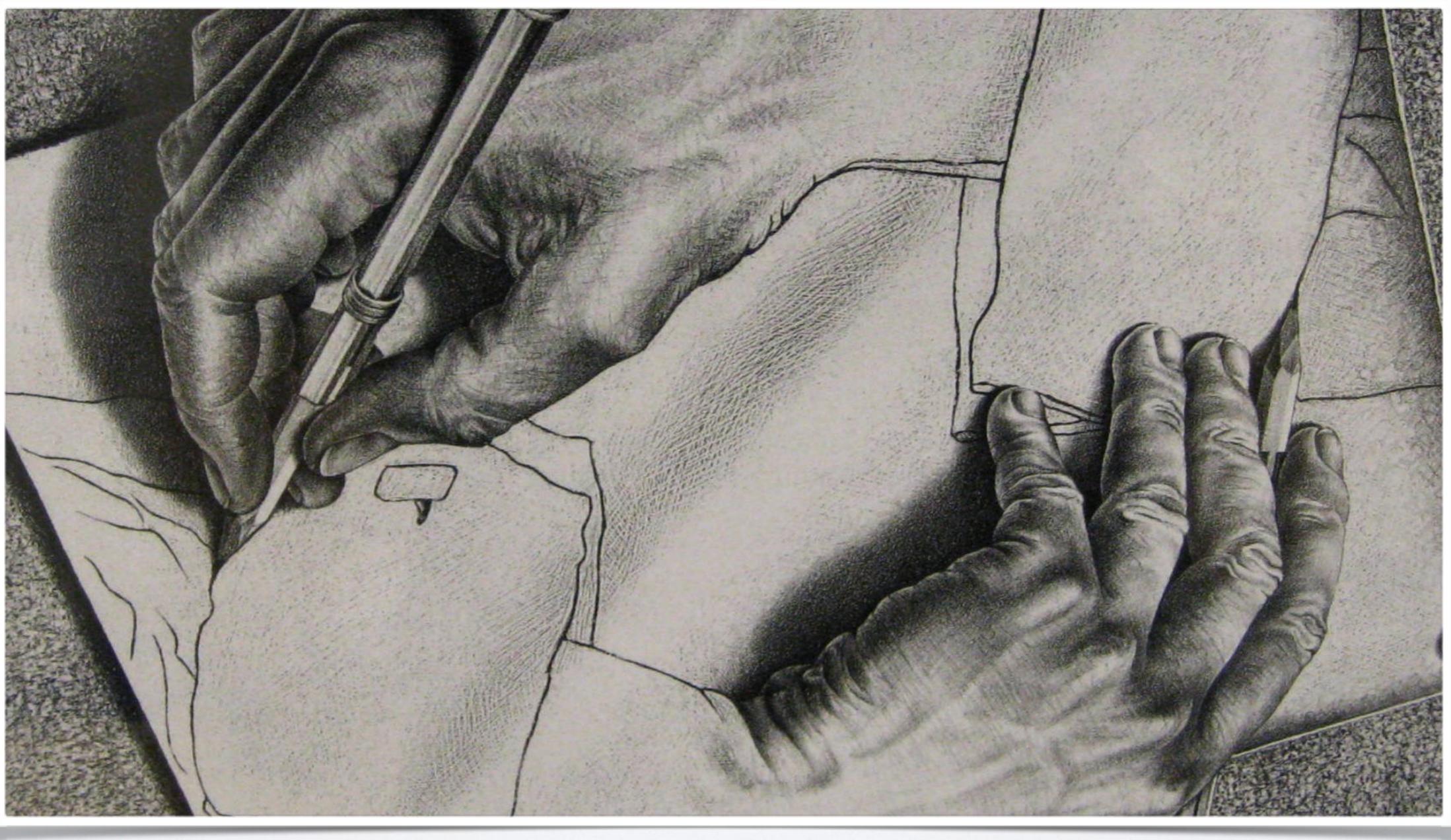
РЕГУЛЯРНЫЕ ВЫРАЖЕНИЯ

Регулярные выражения (англ. [regular expressions](#)) — формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании метасимволов.

(via) [wiki](#)



<http://rubular.com/>



МЕТАПРОГРАММИРОВАНИЕ

Создаем магию вместе с Ruby DSL

БЛОКИ

```
1 def open_file_and_do_smth
2   f = File.open('test')
3   yield(f.read)
4   f.close
5   inside_param = 'Какой блок? Ничего о нем не знаю и знать не хочу'
6 end
7
8 outside_param = 'А я с блоком старые кореша.'
9
10 open_file_and_do_smth do |text|
11   puts text # => Я строка. Мой жизненный цикл – эта презентация.
12   # Вы хоть сочувствуете мне?
13   outside_param += 'Подтверждаю'
14 end
15
16 puts outside_param # => 'А я с блоком старые кореша. Подтверждаю'
17
18 open_file_and_do_smth do |text|
19   puts inside_param # => [NameError] undefined local variable
20   # or method `inside_param` for main:Object
21 end
22
23 puts { :key1 => 'value1', :key2 => 'value2' } # syntax error
24 puts :key1 => 'value1', :key2 => 'value2' # так всё нормально
25
```

```
1 class Person
2   attr_accessor :name, :age,
3                 :position, :ideology
4
5   def initialize
6     yield(self)
7   end
8 end
9
10 spartak = Person.new do |his|
11   his.name = "Alex Mikitenko"
12   his.age = 29
13   his.position = 'qa'
14   his.ideology = 'anarchist'
15 end
16
17
18
19
20
21
22
23
24
25
```

Proc и lambda:

блоки с отложенным выполнением

Proc	lambda
<pre>1 my_proc = Proc.new { puts "Это proc" } 2 puts my_proc 3 # => #<Proc:0x007fbe6b889f10@test.rb:1></pre>	<pre>1 my_lambda = lambda { puts "Это lambda" } 2 my_lambda = -> { puts "Это lambda" } 3 puts my_lambda 4 # => #<Proc:0x007f8b5c1c5658@test.rb:6 (lambda)></pre>

• Игнорирует лишние или недостающие аргументы

• `return` перемещает нас в окружение, где был определен `proc`, и выполняет `return` из него

- Педантично ожидает ровно того количества аргументов, которые было объявлено
- `return` прерывает выполнение блока и возвращает управление в окружение, где была вызвана `lambda`

DSL

Предметно-ориентированный язык
(англ. Domain Specific language,
DSL — «предметно-специфичный
язык») — язык программирования,
специализированный для
конкретной области применения (в
противоположность языку общего
назначения[en], применимому к
широкому спектру областей и не
учитывающему особенности
конкретных сфер знаний)

(via) wiki

```
1 class LoginPage
2   include Watirsome
3
4   text_field :username, label: 'Username'
5   text_field :password, label: 'Password'
6   button :submit_login, text: 'Login'
7
8   def login(username, password)
9     self.username = username
10    self.password = password
11    submit_login
12  end
13 end
14
15 browser = Watir::Browser.new
16 page = LoginPage.new(browser)
17 page.login('demo', 'demo')
18
```

<https://github.com/p0deje/watirsome>

НЕКОТОРЫЕ ДИНАМИЧЕСКИЕ ВКУСНОСТИ

method_missing:

```
1 class MyClass
2   def method_missing(name)
3     puts "#{name} ? Здесь таких нет !"
4   end
5 end
6
7 MyClass.new.undefined_method
8 # => undefined_method ? Здесь таких нет !
```

define_method:

```
1 class MyClass
2   %i[a b c].each do |method|
3     define_method(method) { puts method.to_s }
4   end
5 end
6
7 object = MyClass.new
8 object.a # => a
9 object.b # => b
10 object.c # => c
```

const_set:

```
1 def create_class(name)
2   Object.const_set(name,
3     Class.new do
4       def hello_class
5         "Привет из нового класса!"
6       end
7     end
8   )
9 end
10
11 create_class 'MyClass'
12 MyClass.new.hello_class
13 # => Привет из нового класса!
```

EXCEPTIONS

- обработка исключений выполняется в рамках управляющих конструкций `begin…end`
- перехват исключения производит метод `rescue` или его алиас `fail`
- обязательная для выполнения часть кода идет после метода `ensure`

```
1 begin
2   puts 'value is ' + 5
3   # => выведет в консоль
4   # "[TypeError] no implicit conversion of Fixnum into String"
5   # "А я выполнюсь в любом случае!"
6 rescue TypeError => e
7   puts "[#{e.class}] #{e}"
8 ensure
9   puts "А я выполнюсь в любом случае!"
10 end
```

БИБЛИОТЕКИ (GEMS)

В консоли:

- *gem install rails* => установить конкретный гем
- *gem list* => получить список всех gems
- ...

В приложении:

- *require 'rails'* => подключить rails гем

RBENV, RVM, BUNDLER

- **rvm, rbenv** — это системы управления версиями ruby
- **bundler** — это система управления пакетами (gems) ruby.

Getting Started

Getting started with bundler is easy! Open a terminal window and run this command:

```
$ gem install bundler
```

Specify your dependencies in a Gemfile in your project's root:

```
source 'https://rubygems.org'  
gem 'nokogiri'  
gem 'rack', '~>1.1'  
gem 'rspec', :require => 'spec'
```

СРЕДА РАЗРАБОКИ

```

class PalsController < ApplicationController
  # GET /pals
  # GET /pals.json
  def index
    @pals = Pal.all
    respond_to do |format|
      format.html # index.html.erb
      format.json { render json: @pals }
    end
  end

  # GET /pals/1
  # GET /pals/1.json
  def show
    @pal = Pal.find(params[:id])
    respond_to do |format|
      format.html # show.html.erb
      format.json { render json: @pal }
    end
  end

  # GET /pals/new
  # GET /pals/new.json
  def new
    @pal = Pal.new
    respond_to do |format|
      format.html # new.html.erb
      format.json { render json: @pal }
    end
  end

  # GET /pals/1/edit
end

```

ActiveState & Komodo News

- Stackato: A Great Leap Forward for ActiveState and the Cloud Foundry Platform (Thu, 08 Ma) At ActiveState, we have always worked to make a difference. ...
- What's New in Stackato 1.0 (Wed, 29 Fe) Today's launch of Stackato—the application platform...
- SSH in Stackato (Tue, 28 Fe) A "run" command for Cloud Foundry Cloud Foundry lacks support... [View Details](#)

New Release! Komodo IDE 7

Code faster with the world's fiercest IDE. Upgrade and Save \$50 through March 15!

[What's New](#) [Upgrade](#)

Documentation Community Forums Add-ons

Quick Links

Tutorials and Documentation

Features Ruby on Rails Tutorial XSLT Tutorial Perl Tutorial PHP Tutorial Python Tutorial Regular Expressions Ruby Tutorial

Open Sample Project Contains a number of files showing many of the features of Komodo.

Check Configuration Checks system configuration to ensure all Komodo features are working properly.

Recent Projects and Files

File Path	Last Accessed	Actions
~/flexwage/rails/FlexWage.komodoproject	yesterday	New File...
~/Projects/fiddleevent/Fiddleevent.komodoproject	3 days ago	New Project...
~/Projects/Websites/cpi310/CP 310.komodoproject	11 minutes ago	
~/Projects/Websites/yipplus/Yii Plus.komodoproject	yesterday	
~/Projects/HonCpi310/CP 310 Honors.komodoproject	1 week ago	
~/flexwage/test/FlexWage Test.komodoproject	3 days ago	
~/Projects/fiddleevent/Fiddleevent.komodoproject	does not exist	
~/flexwage/rails/FlexWage Trunk.komodoproject	2 months ago	
~/Projects/Camp-Sparky/members/Camp Sparky Members.komodoproject	3 months ago	
~/Projects/Camp Sparky/members/Camp Sparky Members.komodoproject	does not exist	
~/Sites/campsparky/members/Camp Sparky Members.komodoproject	does not exist	
~/flexwage/trunk/FlexWage Trunk.komodoproject	does not exist	
~/flexwage/importers/FlexWage Importers.komodoproject	does not exist	
~/Sites/campsparky/Camp Sparky Root.komodoproject	4 months ago	
~/Projects/Websites/cpi310/mdl/onlinepharmacy.php	yesterday	
~/flexwage/rails/app/models/employee_observer.rb	1 week ago	
~/Yiip/Controller.php	2 days ago	
User.php	2 days ago	

```

Cuckoo.Event = {
  bind: function(ev, callback) {
    var calls = this._callbacks || (this._callbacks = {});
    var list = calls[ev] || (calls[ev] = []);
    list.push(callback);
    return this;
  },
  unbind: function(ev, callback) {
    var calls;
    if (!ev) {
      this._callbacks = {};
    } else if (calls = this._callbacks) {
      if (!callback) {
        calls[ev] = [];
      } else {
        var list = calls[ev];
        if (!list) return this;
        for (var i = 0, l = list.length; i < l; i++) {
          if (callback === list[i]) {
            list[i] = null;
            break;
          }
        }
      }
    }
    return this;
  },
  fire: function(ev) {
    var calls = this._callbacks[ev];
    var args = Array.prototype.slice.call(arguments, 1);
    var i, len;
    for (i = 0, len = calls.length; i < len; i++) {
      calls[i].apply(this, args);
    }
    return this;
  }
};

Cuckoo.Model = Class.extend({
  storageKey : null,
  ...
});

```

```

1 class GoogleFish
2   attr_accessor :key, :source, :target, :q, :translated_text, :format
3
4   def initialize(key)
5     @key = key
6     @format = :text
7   end
8
9   def translate(source, target, q, options={})
10  @format = :html if options[:html]
11  @source, @target, @q = source, target, q
12  @translated_text = request_translation
13 end
14
15 private
16
17 def request_translation
18   api = GoogleFish::Request.new(self)
19   api.perform_translation
20 end
21 end
22
23 class GoogleFish::Request
24   require 'net/https'
25   require 'addressable/uri'
26   require 'json'
27   attr_accessor :query, :response, :parsed_response
28
29   def initialize(query)
30     @query = query

```

RUBY STYLE GUIDE



RuboCop

<https://github.com/bbatsov/ruby-style-guide>

ВО СЛАБЫ RUBY!



ИЗВЕСНЫЕ ПРОЕКТЫ

- **Github**: без комментариев
- **Basecamp, Redmine**: онлайн-инструменты для управления проектами и задачами
- **Indiegogo**: краудфандинговая платформа

(via) <http://designwebkit.com/>

ИЗВЕСНЫЕ ГЕМЫ

- Rails
- HAML
- Nokogiri
- Cucumber
- Rspec
- Rake
- Bundler
- Capybara
- Watir

ИНКАРНАЦИИ RUBY



mruby



ДОМАШНЕЕ ЗАДАНИЕ

- Сгенерируйте массив со случайными числами и отсортируйте его, не используя метод :sort
 - Создайте объект, у которого будет метод, отличительный от всех остальных объектов этого же класса
-

ПОЛЕЗНЫЕ МАТЕРИАЛЫ

- Изучаем Ruby, Фитцджеральд М.
- Язык программирования Ruby, Флэнаган Д., Мацумото Ю.
- Metaprogramming Ruby 2, Paolo Perrotta
- Введение в Ruby ООП: <http://nashbridges.me/introducing-ruby-oop>
- Модули ruby: <http://habrahabr.ru/post/143990/>