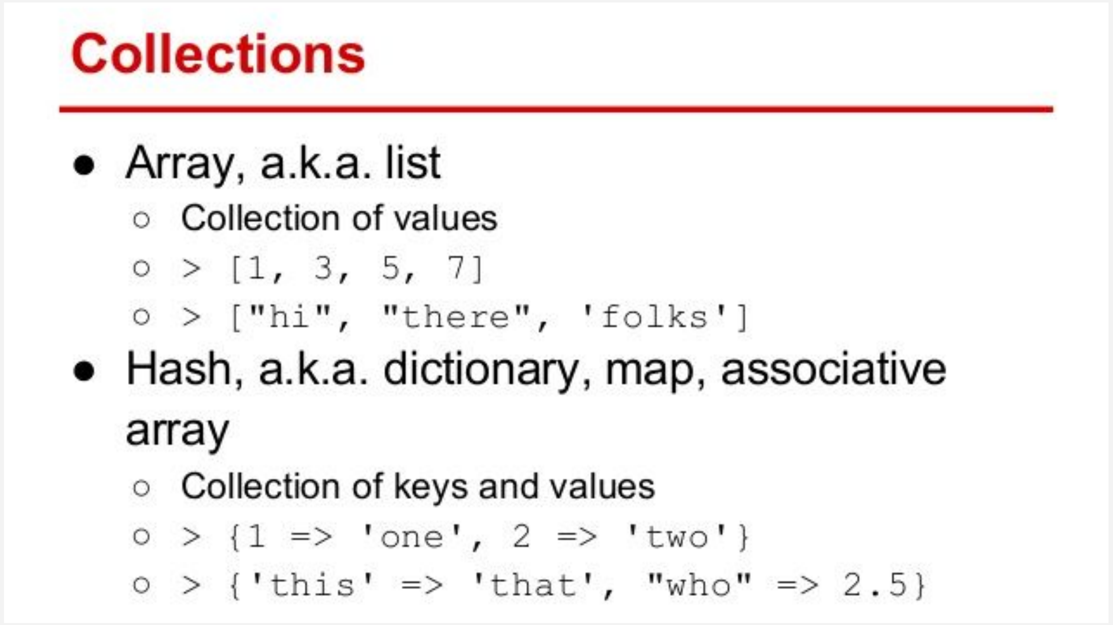
Yalantis

Лекция 6:
Платформа Ruby

Курс лекций по основами web-разработки на языке программирования Ruby

Коллекция

Коллекция - объект, содержащий в себе набор значений одного или различных типов, и позволяющий обращаться к этим значениям.



Collections

- Array, a.k.a. list
  - Collection of values
  - `> [1, 3, 5, 7]`
  - `> ["hi", "there", 'folks']`
- Hash, a.k.a. dictionary, map, associative array
  - Collection of keys and values
  - `> {1 => 'one', 2 => 'two'}`
  - `> {'this' => 'that', "who" => 2.5}`

Массив

```
a = Array.[](1, 2, 3, 4)
b = Array[1,2,3,4]
c = [1,2,3]
d = Array.new(3) {|i| i + 1}
# => [1, 2, 3]
```

```
text_1 = %w[добрый день всем #{"вам".upcase}]
# => ["добрый", "день", "всем", "\#{\"вам\".upcase}"]
text_2  = %W[удачи всем #{"вам".upcase}]
# => ["удачи", "всем", "ВАМ"]
```

```
a = Array.new
# Создать пустой массив
b = Array.new(3)
# [nil, nil, nil]
c = Array.new(3, "Yalantis")
# ["Yalantis", "Yalantis", "Yalantis"]
```

```
c[0].upcase!
# => "YALANTIS"
puts c
# => ["YALANTIS", "YALANTIS", "YALANTIS"]
```

```
a = [1, 2, 3, 4, 5, 6]
b = a[0] # 1
c = a.at(0) # 1
d = a[-2] # 5
e = a.at(-2) # 5
f = a[9] # nil
g = a.at(9) # nil
h = a[3,3] # [ 4, 5, 6]
i = a[2..4]# (3, 4, 5)
j = a[2...4] # [3, 4]
```

```
x = ["Welcome", "to",
     "Yalantis", "school"]
a = x.length # 4
b = x.size #  4
```

```
a = [1, 2, 3, 9, 9]
b = [1, 2, 4, 1, 1]
a <=> b
# => -1
```

```
a > b
# NoMethodError
```

```
a = [1,2]
b = [3,4]
a + b
# => [1,2,3,4]
```

```
a = [1, 2]
b = [3, 4]
a << b
# => [1,2, [3,4]]
```

```
a = [1, 2]
b = [3, 4]
a = a.concat(b) # [1,2,3,4]
```

Массив - основные операции

```
[1,2,3,nil, nil, "b"].compact
# => [1, 2, 3, "b"]

[1,2,21,2,2,3,5].uniq
# => [1, 2, 21, 3, 5]

[1,2,3,4,5,6,7,8,9].reverse
# => [9, 8, 7, 6, 5, 4, 3, 2, 1]

simple_array = %w{Привет я простой массив}
simple_array.reverse_each { |item| print item + " " }
# массив простой я Привет  => ["Привет", "я",
"простой", "массив"]
```

```
x = [1, 5, 9]
x << 1
# => [1, 5, 9, 1]

x = [1, 5, 9]
x.push *[2, 6, 10]
# => => [1, 5, 9, 2, 6, 10]

x.unshift 777
# => [777, 1, 5, 9, 2, 6, 10]

x.pop
# => 10
# [777, 1, 5, 9, 2, 6]

x.shift
# => 777
# [1, 5, 9, 2, 6]
```

```
array = ["Массив", "всему",
"голова"]
array.join(',')
# => "Массив,всему,голова"
array.join(' ')
# => "Массив всему голова"

array = [1,2,3,4,5,6,7,8,9]

array.first
# => 1

array.last
# => 9

array.shuffle
# => [7, 8, 6, 5, 2, 3, 4, 1, 9]
```

И прочие методы из модуля Enumerable

Хеш

```ruby
a1 = Hash.[]("flat", 3, "curved", 2)
a2 = Hash.[]("flat"=>3, "curved"=>2)

b1 = Hash["flat",3,"curved",21]
b2 = Hash["flat"=>3,"curved"=>2]

c1 = {"flat": 3, "curved": 21}
c2 = {"flat"=>3,"curved"=>21}

d = Hash.new

e = Hash.new(99)
e[:p]
# => 99

f = Hash.new("a"=>3)
f[:s]
# => {"a"=>3}
```

```ruby
a = {}
a["flat"] = 3
a.[]=("curved",2)
a.fetch("flat") # 3
a.[]("flat") # 3
a["flat"] # 3
a["bent"] # nil

{}[:some_key]
# => nil

{}.fetch(:some_key)
# => KeyError

a = {"a"=>1, "b"=>2}
a.has_key? "c" # false
a.include? "a" # true
a.key? 2 # false
a.member? "b" # true

{one_day: :another}.invert
```

```ruby
pairs = [[2, 3],[4, 5],[6,7]]
array = [2, 3, 4, 5, 6, 7]
h1 = pairs.to_h
# => {2=>3, 4=>5, 6=>7}
h2 = Hash[pairs]
# => {2=>3, 4=>5, 6=>7}
h3 = Hash[*array]
# => {2=>3, 4=>5, 6=>7}

h3.delete(2)
# => 3
puts h3
# => {4=>5, 6=>7}

{first: 1, second: 2}.each do |key, value|
 puts key
 puts value
end
# first
# 1
# second
# 2
```

Множество

```
require 'set'

products = Set.new

products << 1
products << 1
products << 2

products
# => #<Set: {1, 2}>
```

```
products.include?(1)
# true

products[0]
# undefined method `[]'

products.to_a
# [1, 2]

Set.new(1..3) & Set.new(2..5)
# Set: {2, 3}

Set.new(1..3) & Set.new(2..5)
# Set: {2, 3}

Set.new(25..27) <= Set.new(20..30)
# true
```

```
require 'set'
s1 = Set[1, 2]                  #=> #<Set: {1, 2}>
s2 = [1, 2].to_set              #=> #<Set: {1, 2}>
s1 == s2                        #=> true
s1.add("foo")                   #=> #<Set: {1, 2, "foo"}>
s1.merge([2, 6])                #=> #<Set: {1, 2, "foo", 6}>
s1.subset?(s2)                  #=> false
s2.subset?(s1)                  #=> true
```

```
Set[2,3,1] == Set[3,1,2]
# true
```

Упорядоченное множество - SortedSet

Если вы хотите чтобы ваши наборы оставались отсортированными вы можете использовать класс SortedSet

Два условия для его использования:

1. Объекты которые вы добавляете должны иметь реализованный метод <=>
2. Объекты должны быть сравнимы друг с другом (числа к числам, строки к строкам)

```
sorted_numbers = SortedSet.new

sorted_numbers << 5
sorted_numbers << 2
sorted_numbers << 1

sorted_numbers
# SortedSet: {1, 2, 5}

sorted_numbers << """"
# ArgumentError (comparison of Integer with Hash failed)
```

Упорядоченное множество - SortedSet

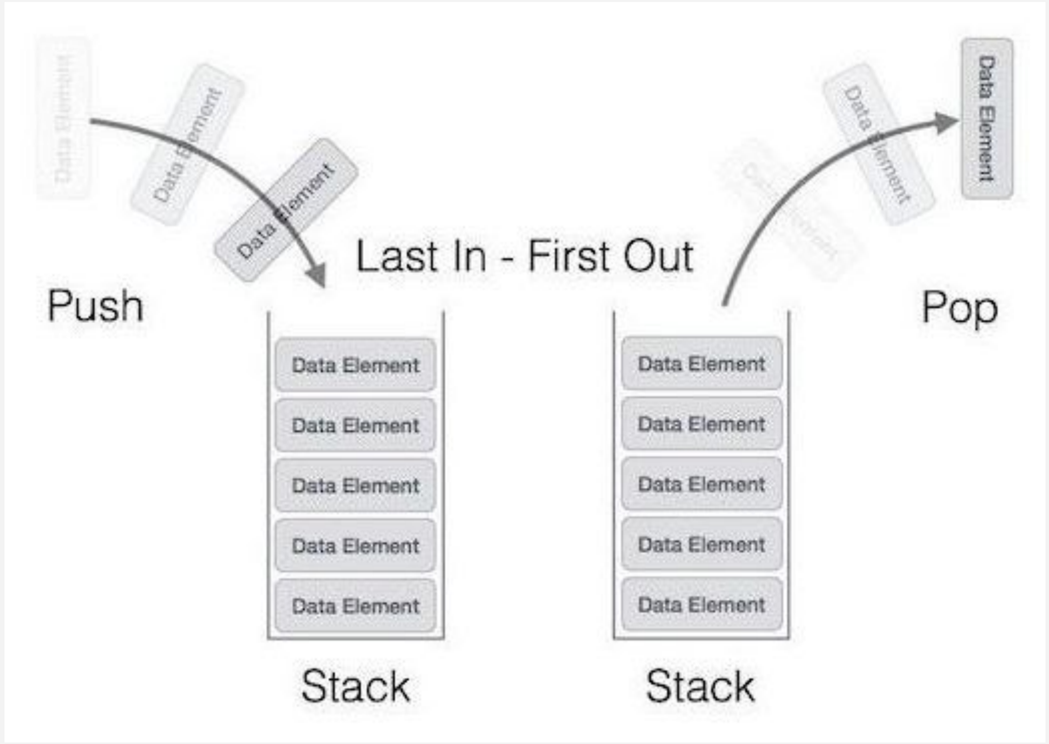
Если вы хотите чтобы ваши наборы оставались отсортированными вы можете использовать класс SortedSet

Два условия для его использования:

1. Объекты которые вы добавляете должны иметь реализованный метод <=>
2. Объекты должны быть сравнимы друг с другом (числа к числам, строки к строкам)

```
sorted_numbers = SortedSet.new

sorted_numbers << 5
sorted_numbers << 2
sorted_numbers << 1

sorted_numbers
# SortedSet: {1, 2, 5}

sorted_numbers << ""
# ArgumentError (comparison of Integer with Hash failed)
```

Стек



Реализуем стек

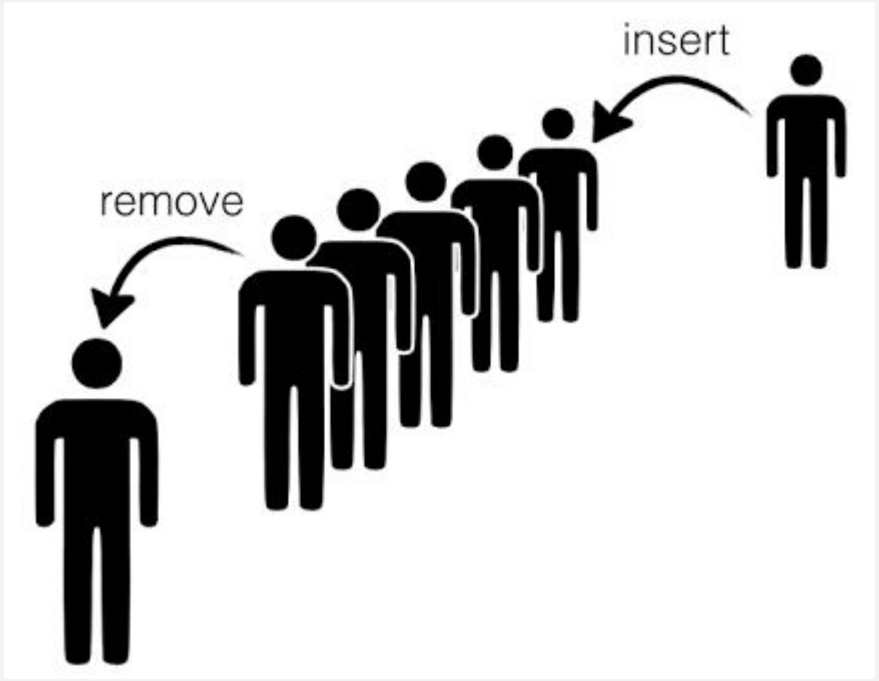
```ruby
class Stack
 def initialize
   @store = []
 end

 def push(x)
   @store.push x
 end

 def pop
   @store.pop
 end

 def peek
   @store.last
 end

 def empty?
   @store.empty?
 end
end
```

Задача на проверку скобок

```
class BracketMaster
 attr_accessor :stack, :string, :open_brackets, :close_brackets

 def initialize(string)
   self.stack = Stack.new
   self.string = string
   self.open_brackets = ['{','<','(','[']
   self.close_brackets = ['}','>',')',']']
 end

 def check
   string.each_char do |char|
     if open_brackets.include?(char)
       stack.push(char)
     elsif close_brackets.include?(char)
       open_brackets.index(stack.peek) != close_brackets.index(char) ?
         (return false) : stack.pop
     end
   end

   return stack.empty?
 end
end
```

```
BracketMaster.new("{hello: {world: [1,2,3]}").check
# => false

BracketMaster.new("{hello: {world: [1,2,3]}}").check
# => true
```

Очередь



Реализуем очередь

```
class Queue
 def initialize
   @store
 end
 def enqueue(x)
   @store << x
 End

 def dequeue
   @store.shift
 end

 def peek
   @store.first
 end

 def length
   @store.length
 end

 def empty?
   @store.empty?
 end
end
```

Встроенная очередь

```ruby
queue = Queue.new

producer = Thread.new do
 5.times do |i|
   sleep rand(i) # simulate expense
   queue << i
   puts "#{i} produced"
 end
end

consumer = Thread.new do
 5.times do |i|
   value = queue.pop
   sleep rand(i/2) # simulate expense
   puts "consumed #{value}"
 end
end
```

```ruby
que = SizedQueue.new(5)
```

Диапазон

```
('a'..'z').each {|i| print i }
# => abcdefghijklmnopqrstuvwxyz
(3..6).each {|i| print i }
# => 3456
(3...6).each {|i| print i }
# => 345

r1 = 3..6
r2 = 3...6
r1.first
# 3
r2.last
# 6

r1.include?(5)
# true
r2.include?(100)
# false
```

```
(10..20).step(2).to_a
# [10, 12, 14, 16, 18, 20]


require 'time'
t1 = DateTime.new
t2 = DateTime.new + 30
next_30_days = t1..t2
next_30_days.select(&:friday?).map(&:day)
# => [5, 12, 19, 26]
```

Диапазон - что внутри ?

```ruby
class LetterMultiplier
 include Comparable

 attr_reader :count

 def initialize(letter, count)
   @letter = letter
   @count  = count
 end

 def succ
   self.class.new(@letter, @count + 1)
 end

 def <=>(other)
   count <=> other.count
 end
end

a = LetterMultiplier.new('w', 2)
b = LetterMultiplier.new('w', 8)
```

```ruby
puts Array(a..b)
```

```
#<LetterMultiplier:0x00007f989a109200>
#<LetterMultiplier:0x00007f989a227d80>
#<LetterMultiplier:0x00007f989a227d30>
#<LetterMultiplier:0x00007f989a227d08>
#<LetterMultiplier:0x00007f989a227c68>
#<LetterMultiplier:0x00007f989a227c40>
#<LetterMultiplier:0x00007f989a227b28>
```

Дата и время

STANDARD TIME ZONES OF THE WORLD
ARCTIC OCEAN
Greenland
CANADA
U.S.
UNITED STATES
MEXICO
NORTH PACIFIC OCEAN
NORTH ATLANTIC OCEAN
SOUTH PACIFIC OCEAN
SOUTH ATLANTIC OCEAN
BRAZIL
COLOMBIA
PERU
BOLIVIA
ARGENTINA
CHILE
RUSSIA
CHINA
INDIA
MONGOLIA
KAZAKHSTAN
TURKEY
IRAN
SAUDI ARABIA
ALGERIA
LIBYA
EGYPT
NIGER
CHAD
SUDAN
INDONESIA
AUSTRALIA
NEW ZEALAND
INDIAN OCEAN
Scale 1:85,000,000 at 0°
Miller Cylindrical Projection
Coordinated Universal Time (UTC) formerly Greenwich Mean Time (GMT)
WEST
EAST

Time

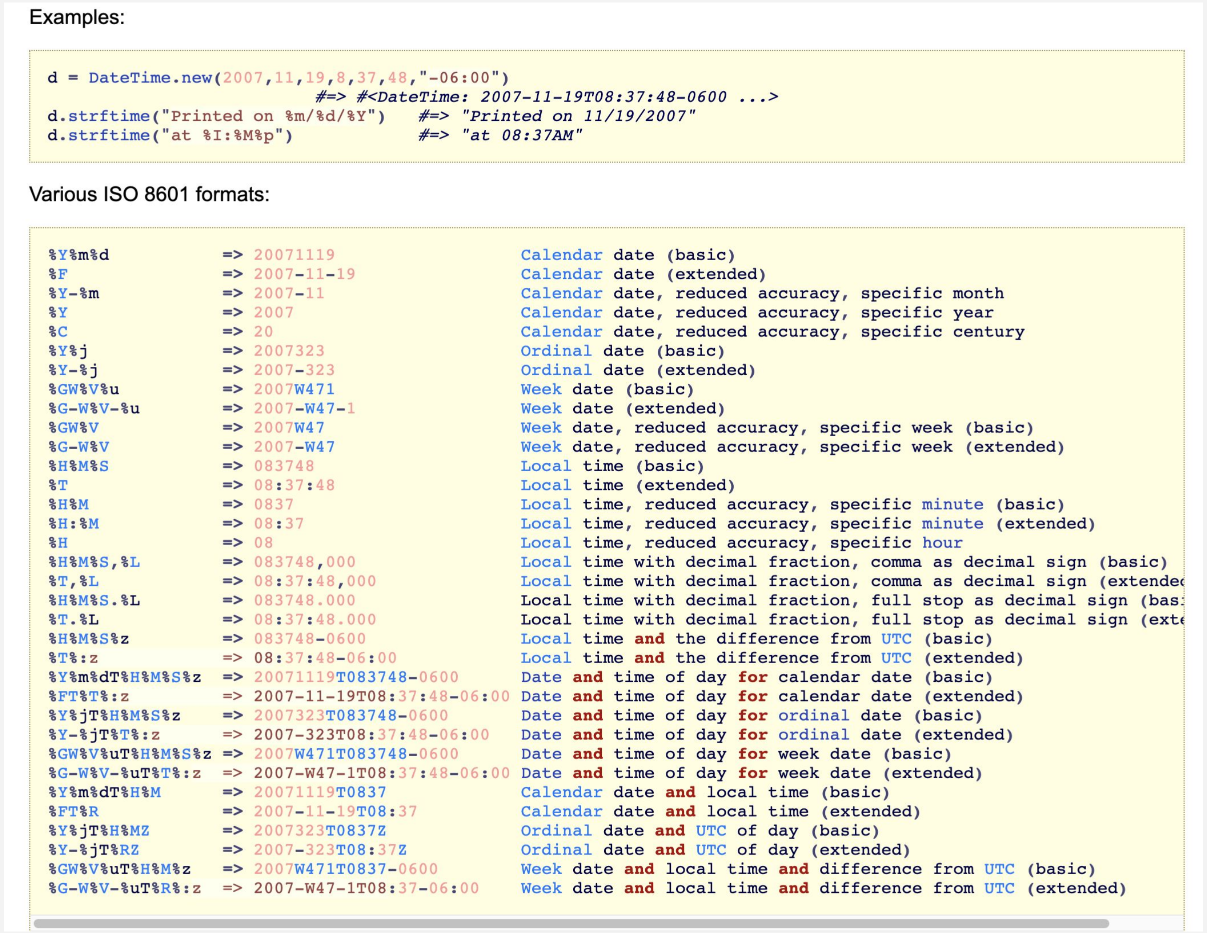
```
Time.now.to_i
# 1587405049

time = Time.new + 10

Time.new > time
# false

Time.now - 86400
# 2020-04-19 20:55:00 +0300

Time.now - 86400 * 100000
# 1746-07-06 20:56:11 +0300

t = Time.now
t.zone
# "EEST"
t.utc_offset / 3600
# 3 - Смещение часовой зоны
```

```
Time.now
# 2020-04-19 15:43:20 +0300
Time.new(2018, 1, 1)
# 2018-01-01 00:00:00 +0300
Time.at(15000000000)
# 2445-05-01 05:40:00 +0300
Time.now.utc
# 2020-04-20 17:58:17 UTC
```

```
t = Time.now
puts t.day
# 19
puts t.month
# 4
puts t.hour
# 15

t = Time.now
puts t.monday?
# false
puts t.sunday?
# true
puts t.friday?
# false
```

strftime

```ruby
time = Time.new
time.strftime("%d/%m/%Y")
# "20/04/2020"
time.strftime("%k:%M")
# "20:44"
time.strftime("%I:%M %p")
# "08:44 PM"
time.strftime("Today is %A")
# "Today is Monday"
time.strftime("%d of %B, %Y")
# "20 of April, 2020"
time.strftime("Unix time is %s")
# "Unix time is 1587404691"
```



Examples:

```
d = DateTime.new(2007,11,19,8,37,48,"-06:00")
                                 #=> #<DateTime: 2007-11-19T08:37:48-0600 ...>
d.strftime("Printed on %m/%d/%Y")    #=> "Printed on 11/19/2007"
d.strftime("at %I:%M%p")             #=> "at 08:37AM"
```

Various ISO 8601 formats:

```
%Y%m%d           => 20071119                  Calendar date (basic)
%F               => 2007-11-19                Calendar date (extended)
%Y-%m            => 2007-11                   Calendar date, reduced accuracy, specific month
%Y               => 2007                      Calendar date, reduced accuracy, specific year
%C               => 20                        Calendar date, reduced accuracy, specific century
%Y%j             => 2007323                   Ordinal date (basic)
%Y-%j            => 2007-323                  Ordinal date (extended)
%GW%V%u          => 2007W471                  Week date (basic)
%G-W%V-%u        => 2007-W47-1                Week date (extended)
%GW%V            => 2007W47                   Week date, reduced accuracy, specific week (basic)
%G-W%V           => 2007-W47                  Week date, reduced accuracy, specific week (extended)
%H%M%S           => 083748                    Local time (basic)
%T               => 08:37:48                  Local time (extended)
%H%M             => 0837                      Local time, reduced accuracy, specific minute (basic)
%H:%M            => 08:37                     Local time, reduced accuracy, specific minute (extended)
%H               => 08                        Local time, reduced accuracy, specific hour
%H%M%S,%L        => 083748,000                Local time with decimal fraction, comma as decimal sign (basic)
%T,%L            => 08:37:48,000              Local time with decimal fraction, comma as decimal sign (extende
%H%M%S.%L        => 083748.000                Local time with decimal fraction, full stop as decimal sign (bas
%T.%L            => 08:37:48.000              Local time with decimal fraction, full stop as decimal sign (exte
%H%M%S%z         => 083748-0600               Local time and the difference from UTC (basic)
%T%:z            => 08:37:48-06:00            Local time and the difference from UTC (extended)
%Y%m%dT%H%M%S%z  => 20071119T083748-0600      Date and time of day for calendar date (basic)
%FT%T%:z         => 2007-11-19T08:37:48-06:00 Date and time of day for calendar date (extended)
%Y%jT%H%M%S%z    => 2007323T083748-0600       Date and time of day for ordinal date (basic)
%Y-%jT%T%:z      => 2007-323T08:37:48-06:00   Date and time of day for ordinal date (extended)
%GW%V%uT%H%M%S%z => 2007W471T083748-0600      Date and time of day for week date (basic)
%G-W%V-%uT%T%:z  => 2007-W47-1T08:37:48-06:00 Date and time of day for week date (extended)
%Y%m%dT%H%M      => 20071119T0837             Calendar date and local time (basic)
%FT%R            => 2007-11-19T08:37          Calendar date and local time (extended)
%Y%jT%H%MZ       => 2007323T0837Z             Ordinal date and UTC of day (basic)
%Y-%jT%RZ        => 2007-323T08:37Z           Ordinal date and UTC of day (extended)
%GW%V%uT%H%M%z   => 2007W471T0837-0600        Week date and local time and difference from UTC (basic)
%G-W%V-%uT%R%:z  => 2007-W47-1T08:37-06:00    Week date and local time and difference from UTC (extended)
```

Date

```
Date.today
# <Date: 2020-04-20 ((2458960j,0s,0n),+0s,2299161j)>
Date.new
# <Date: -4712-01-01 ((0j,0s,0n),+0s,2299161j)>

(Date.today + 100).to_s
# "2020-07-29"

Date::MONTHNAMES
# [nil, "January", "February", "March",
#  "April", "May", "June", "July",
#  "August", "September", "October",
#  "November", "December"]

Date::DAYNAMES
# ["Sunday", "Monday", "Tuesday",
#   "Wednesday", "Thursday",
#   "Friday", "Saturday"]

Date::DAYNAMES.rotate(1)
# ["Monday", "Tuesday", "Wednesday",
#  "Thursday", "Friday", "Saturday",
#  "Sunday"]
```

```
Date.parse("10/10/2010")
# -> 2010-10-10
Date.parse("September 3")
# -> 2020-09-03
Date.parse("May I have a cup of coffee, please?")
# -> "2020-05-01"

Date.iso8601("2000-01-01")
# -> "2000-01-01"
# Строгий формат: year-month-day
Date.parse("May i help you ?")
# -> "2020-05-01"
Date.iso8601("May i help you ?")
# => ArgumentError (invalid date)

Date.strptime("3 of September", "%d of %B")
# -> "2020-09-03"
```

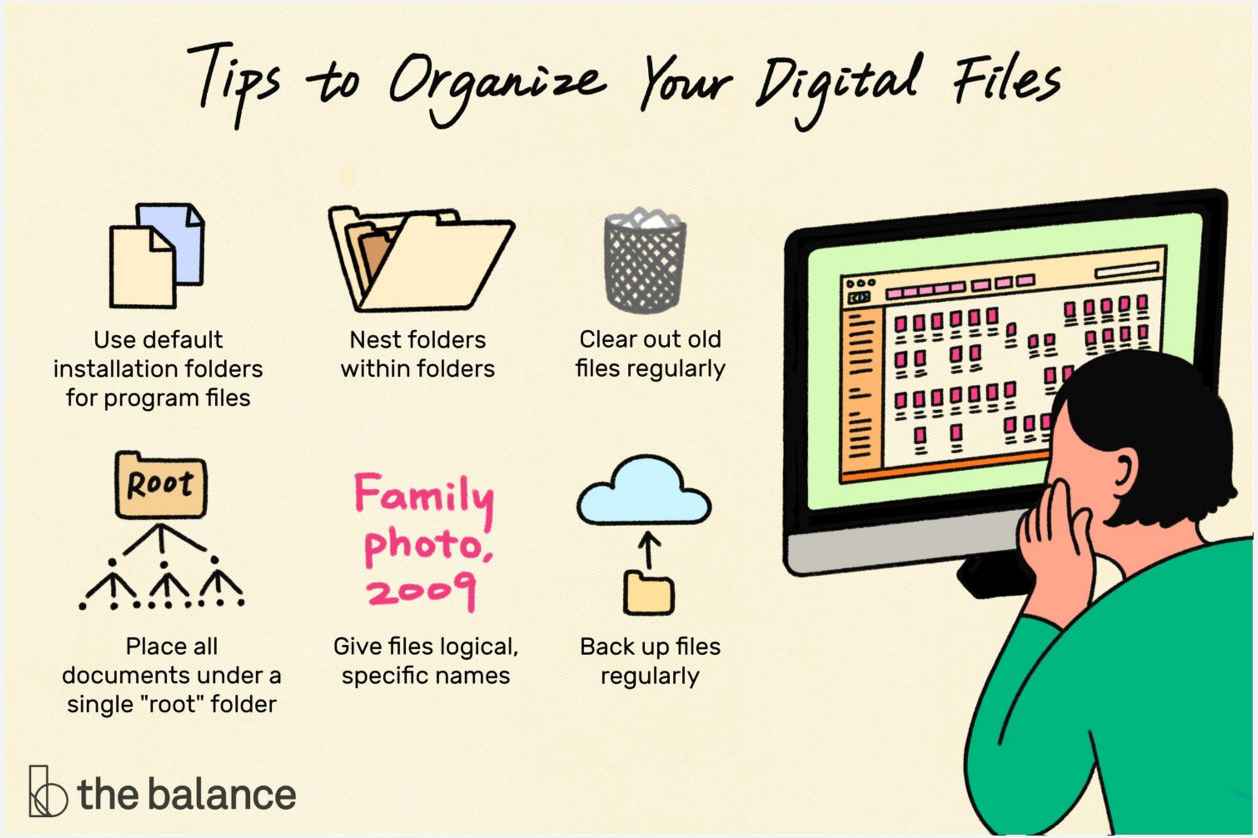
DateTime

DateTime выполняет ту же работу что и Time, с основным отличием в том, что Time реализован на C, поэтому он будет быстрее.

```
require 'date'
DateTime.superclass
# Date
DateTime.now.to_s
# "2020-04-20T21:20:24+03:00"
```

```
Comparison:
  Time:        2644596.6 i/s
  DateTime:   231634.8 i/s - 11.42x  slower
```

Работа с файлами


Tips to Organize Your Digital Files
Use default installation folders for program files
Nest folders within folders
Clear out old files regularly
Root
Place all documents under a single "root" folder
Family photo, 2009
Give files logical, specific names
Back up files regularly
the balance

File

```ruby
file = File.open("welcome.txt")
# => #<File:welcome.txt>
file_data = file.read
# => "Hello\nYalantis\nCourse\n"
file_data = file.readlines.map(&:chomp)
# => ["Hello", "Yalantis", "Course"]
file.close
# => true
file
# => #<File:welcome.txt (closed)>
```

```ruby
file_data = File.read("welcome.txt").split
# => ["Hello", "Yalantis", "Course"]
```

```ruby
File.foreach("welcome.txt") { |line| puts line }
# Hello
# Yalantis
# Course
```

```ruby
File.open("log.txt", "w") { |f| f.write "#{Time.now} - User logged in\n" }
```

```ruby
File.write("log.txt", "data...")
File.write("log.txt", "data...", mode: "a")
File.write("log.txt", [1,2,3].join("\n"), mode: "a")
```

Не забывайте закрывать за собой File

```ruby
file = File.open("some_text.txt", "w")
file.puts "Строка 1"
file.puts "Строка 2"
file.puts "Строка которая завершает текст"

File.foreach("some_text.txt") { |line| puts line }
# => nil
```

```ruby
file = File.open("random_text.txt", "w")
file.puts "Строка 1"
file.puts "Строка 2"
file.puts "Строка которая завершает текст"
file.close
File.foreach("random_text.txt") { |line| puts line }
# Строка 1
# Строка 2
# Строка которая завершает текст
```

File

```ruby
# Переименование файла
File.rename("old-name.txt", "new-name.txt")
# Размер файла в байтах
File.size("users.txt")
# Существует ли файл ?
File.exists?("log.txt")
# Получить расширение файла,
# работает даже если файла не существует
File.extname("users.txt")
# => ".txt"
# Получить имя файла без директории
File.basename("/tmp/ebook.pdf")
# => "ebook.pdf"
# Получить путь без имени файла
File.dirname("/tmp/ebook.pdf")
# => "/tmp"
# Это директория ?
File.directory?("cats")
```

```ruby
File.stat("log.txt")
# => #<File::Stat dev=0x1000004, ino=34504486,
# mode=0100644, nlink=1, uid=501, gid=20,
# rdev=0x0, size=19, blksize=4096, blocks=8,
# atime=2020-04-20 23:36:22 +0300, mtime=2020-04-20
23:36:20 +0300,
# ctime=2020-04-20 23:36:20 +0300, birthtime=2020-04-20
23:35:22 +0300>
```

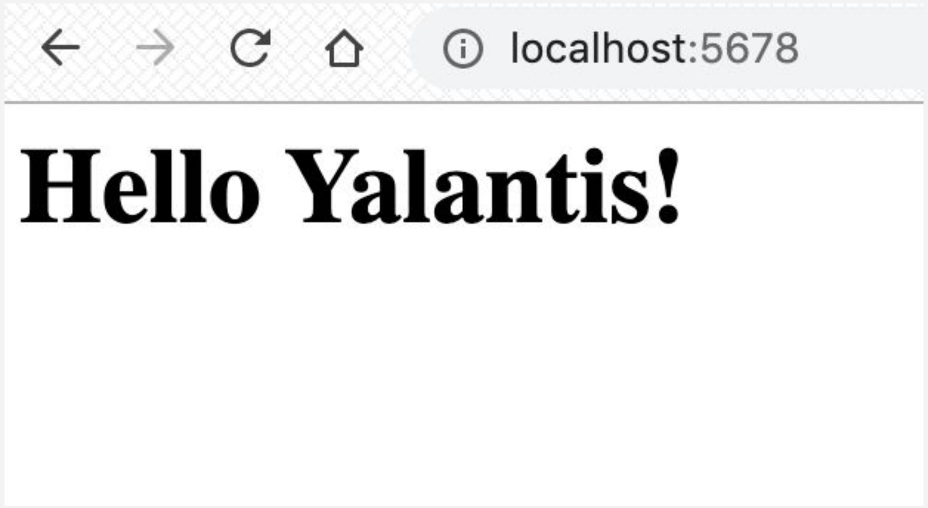
Dir и FileUtils

```
# Все файлы в текущей директории
Dir.glob("*")
# Все файлы содержащие "spec" в имени
Dir.glob("*spec*")
# Все руби файлы
Dir.glob("*.rb")
# Вывести путь
Dir.pwd

Dir.empty?("/tmp")
# false

Dir.exists?("/home/den")
# false

# Создать директорию
Dir.mkdir("/tmp/testing")
```

```
require 'fileutils'

FileUtils.cd(dir, **options)
FileUtils.cd(dir, **options) {|dir| block }
FileUtils.pwd()
FileUtils.mkdir(dir, **options)
FileUtils.mkdir(list, **options)
FileUtils.mkdir_p(dir, **options)
FileUtils.mkdir_p(list, **options)
FileUtils.rmdir(dir, **options)
FileUtils.rmdir(list, **options)
FileUtils.ln(target, link, **options)
FileUtils.ln(targets, dir, **options)
FileUtils.ln_s(target, link, **options)
FileUtils.ln_s(targets, dir, **options)
FileUtils.ln_sf(target, link, **options)
FileUtils.cp(src, dest, **options)
FileUtils.cp(list, dir, **options)
FileUtils.cp_r(src, dest, **options)
FileUtils.cp_r(list, dir, **options)
```

```
FileUtils.mv(src, dest, **options)
FileUtils.mv(list, dir, **options)
FileUtils.rm(list, **options)
FileUtils.rm_r(list, **options)
FileUtils.rm_rf(list, **options)
FileUtils.install(src, dest, **options)
FileUtils.chmod(mode, list, **options)
FileUtils.chmod_R(mode, list, **options)
FileUtils.chown(user, group, list, **options)
FileUtils.touch(list, **options)
```

HTTP-сервер


localhost:5678

Hello Yalantis!

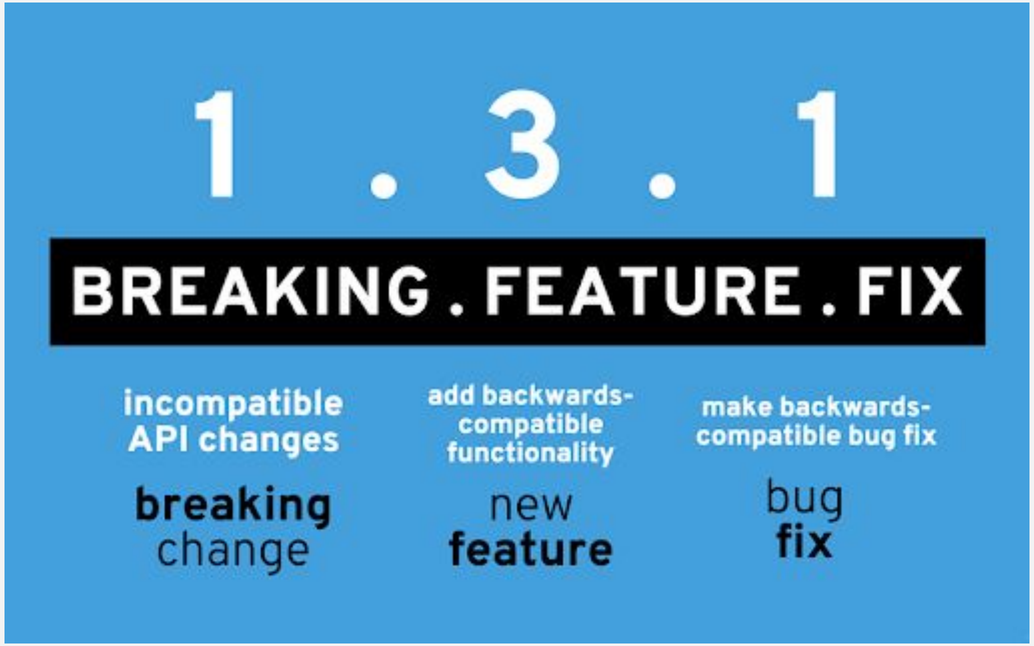
```
require 'socket'
server = TCPServer.new(5678)

while session = server.accept
 request = session.gets
 puts request

 session.print "HTTP/1.1 200\r\n" # 1
 session.print "Content-Type: text/html\r\n" # 2
 session.print "\r\n" # 3
 session.print "<html><body><h1>Hello Yalantis!</h1></body></html>\n" #4

 session.close
end
```

HTTP-запрос

```
require "socket"
require "time"
resp = TCPSocket.new("time.nist.gov", 13).read
time = resp.split(" ")[2] + " UTC"
remote = Time.parse(time)

puts "Локальное: #{Time.now.utc.strftime("%H:%M:%S") }"
# Локальное: 19:35:30
puts "Удаленное: #{remote.strftime("%H:%M:%S")}"
# Удаленное: 19:35:31
```

Семантическое версионирование

```
gem "supergem", "~> 1.0"
# 1.0.2 -> 1.0.3
# bundle update supergem
```



Что почитать ?

1. https://www.tutorialspoint.com/ruby/ruby_date_time.htm
2. https://code.tutsplus.com/tutorials/ruby-for-newbies-working-with-directories-and-files--net-18810
3. https://semver.org/lang/ru/
4. Главы 3.3 - 3.5, 9 книги “Язык программирования Ruby”

Yalantis

Спасибо!

Остались вопросы? Буду рад вам ответить. Не забывайте пользоваться учебным чатом