2013年2月2日

Ruby初級者向けレッスン 44回
― ブロック ―

ひがき @ Ruby関西

お品書き

- ブロックとは?
  - 繰り返し
  - ハリウッドの原理
- メソッドにブロックを渡す
- ブロックで値を受け取る
- メソッドでブロックを受け取る
- ブロックに値を渡す

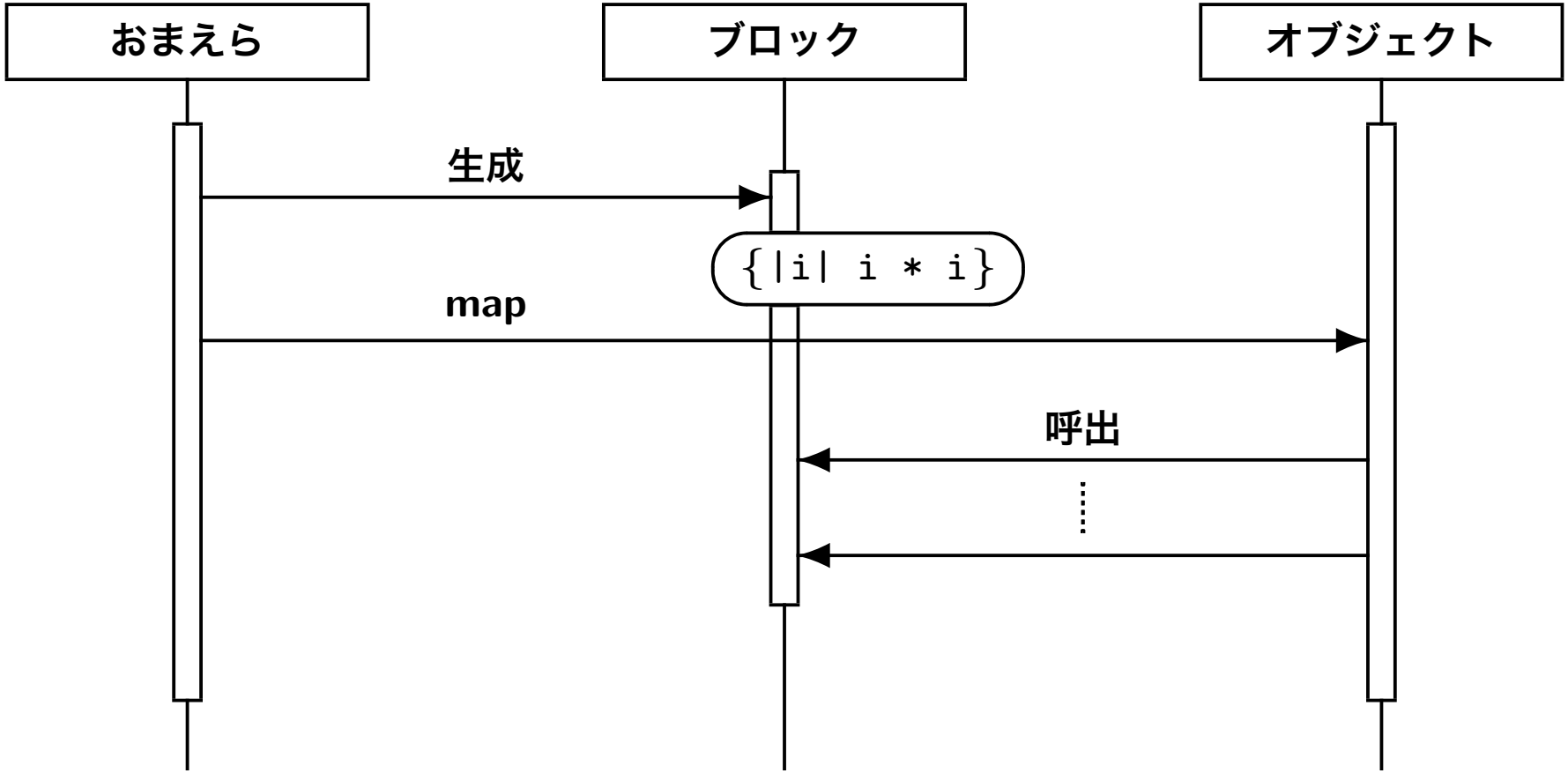
繰り返し

```
a = [0, 1, 2]
a.each do |i|
  puts i
end

a.each{|i| puts i}
# >> 0
# >> 1
# >> 2
```

便利な例

```
a = [0, 1, 2, 3]        # => [0, 1, 2, 3]

a.map{|i| i * i}        # => [0, 1, 4, 9]
a.select{|i| i.even?}       # => [0, 2]
a.inject{|s, i| s + i}      # => 6
a.find{|i| i.odd?}          # => 1
a.all?{|i| i.even?}         # => false
a.any?{|i| i.even?}         # => true
```

array.map{|i| i * i}



ブロックを渡す

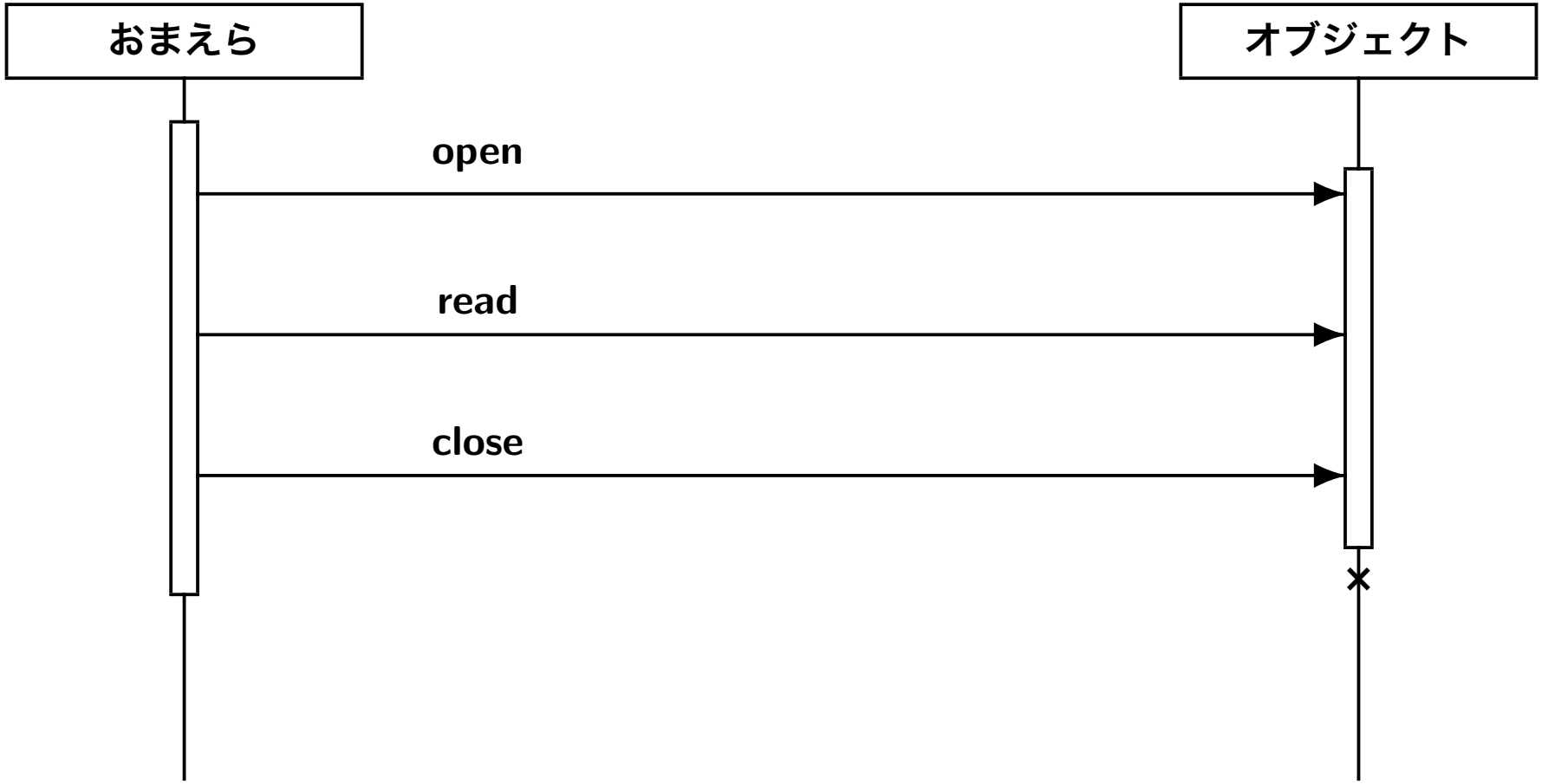
- メソッドには、ブロックをひとつ渡せる。
- ブロックをどう使うかは、メソッド次第。
  - 繰り返し
  - ハリウッドの原理

```
open('hello.txt') # => #<File:hello.txt>
open('hello.txt'){|f| f.read}
                          # => "こんにちは\n"
```

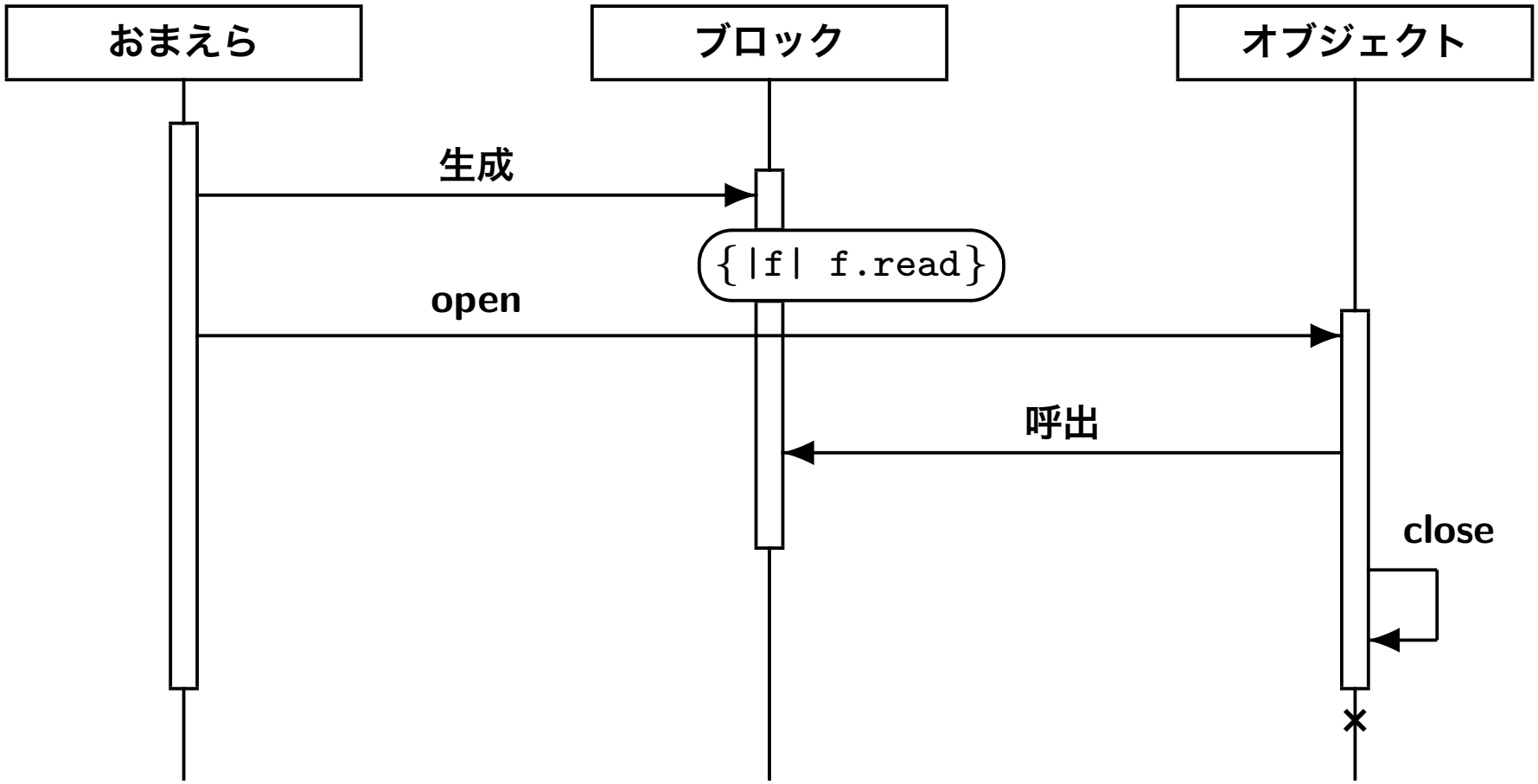
ハリウッドの原理

```
# open('hello.txt'){|f| f.read}
begin
  f = open('hello.txt')
  f.read
ensure
  f.close unless f.nil?
end
```

ブロックのない open



ブロック付き open



値を受け取る

- ブロックは、値を受け取れる。(多重代入)
- 何を幾つ受け取れるかは、メソッド次第。
- 参考
  - 第54回 Ruby/Rails勉強会@関西
    Ruby初級者向けレッスン 42回
    繰り返しと多重代入

値を受け取る (2)

- 受け取るか受け取らないかは、ブロック次第。

```
2.times{puts 'こんにちは'}
# >> こんにちは
# >> こんにちは

2.times{|i| puts i}
# >> 0
# >> 1
```

Hash の例

```
people = {matz: 47, dhh: 32}
                # => {:matz=>47, :dhh=>32}

people.each{|person| p person}

# >> [:matz, 47]
# >> [:dhh, 32]
```

Hash の例 (2)

```
people = {matz: 47, dhh: 32}

people.each do |name, age|
  p "#{name}(#{age})"
end

# >> "matz(47)"
# >> "dhh(32)"
```

each_cons の例

```
midosuji = ["梅田", "淀屋橋",
            "本町", "心斎橋", "なんば"]

midosuji.each_cons(2){|path| p path}

# >> ["梅田", "淀屋橋"]
# >> ["淀屋橋", "本町"]
# >> ["本町", "心斎橋"]
# >> ["心斎橋", "なんば"]
```

each_cons の例 (2)

```
midosuji.each_cons(2) do |from, to|
  p "#{from} - #{to}"
end

# >> "梅田 - 淀屋橋"
# >> "淀屋橋 - 本町"
# >> "本町 - 心斎橋"
# >> "心斎橋 - なんば"
```

each_cons の例 (3)

```
a = [*0..3]       # => [0, 1, 2, 3]
a.each_cons(3){|i| p i}
# >> [0, 1, 2]
# >> [1, 2, 3]

a.each_cons(3){|i, j| p [i, j]}
# >> [0, 1]
# >> [1, 2]
```

おかしいな? と思ったら

```
p unknowns.first
                # >> [1, ["matz", 47]]

unknowns.each do |id, (name, age)|
  id                  # => 1
  name                # => "matz"
  age                 # => 47
    ...
end
```

ブロックを受け取るメソッド

- こんな感じで呼びたい

```
monta{puts 'block!'}


# >> block!
# >> block!
# >> 大切なことなので
```

ブロックを受け取る

```
def monta
  yield
  yield
  puts ’大切なことなので’
end
```

ブロックを受け取る (2)

```
def monta(&block)
  block.call
  block.call
  puts '大切なことなので'
end
```

値を渡す

```
def monta
  yield ’大切なことなので’
  yield ’大切なことなので’
end

monta{|i| puts "#{i} block!"}

# >> 大切なことなので block!
# >> 大切なことなので block!
```

値を渡す (2)

```
def monta(&block)
  block.call '大切な', 'ことなので'
  block.call ['大切な', 'ことなので']
end

monta{|i| puts "#{i} block!"}

# >> 大切な block!
# >> ["大切な", "ことなので"] block!
```

ブロックは Proc

```
block = Proc.new do |i, j|
  puts "#{i}#{j} block!"
end

monta(&block)

# >> 大切なことなので block!
# >> 大切なことなので block!
```

まとめ

- ブロックとは?
  - 繰り返し
  - ハリウッドの原理
- メソッドにブロックを渡す
- ブロックで値を受け取る
- メソッドでブロックを受け取る
- ブロックに値を渡す

演習問題 0

今日のレッスンで分からなかったこと、疑問に思ったことをグループで話し合ってみよう。

演習問題 1

0 から 9 までの数値をもつ配列 a がある。

- 各要素を順番に表示しよう。
- 各要素を 2倍した値を持つ配列を作ろう。
- 全要素の合計値を計算しよう。

```
a = (0..9).to_a
a    # => [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

演習問題 2

0 から 9 までの数値をもつ配列 a がある。

- 奇数の要素だけを持つ配列を作ろう。
- ただし odd? メソッドは使用禁止。

演習問題 3

Enumerable#map を自作してみよう。

```
module Enumerable
  def my_map
    ......
  end
end
```

ただし Enumerable#map と Enumerable#map! は使用禁止。

自己紹介

- 名前 (ニックネーム)
- 普段の仕事・研究内容・代表作
- Ruby歴・コンピュータ歴
- 勉強会に来た目的
- などなど