

# **Ruby on Rails 講義**

## **第18回, 第19回 Rails基礎**

**Kuniaki IGARASHI/igaiga**

**2012.10.18, 25 at 一橋大学**

**社会科学における情報技術とコンテンツ作成IV**  
**(ニフティ株式会社寄附講義)**

○ 剰余金の配当に関するお知らせ

○ ニフティ、「@nifty EMOBILE LTE 定額にねんプラン」の提供を開...

○ 「@nifty温泉」で「母の日 全国一斉！100のありがとう風呂」特設サイト公...

○ 「スマブレ！」のサービス停止について

○ ニフティとサンリオウエーブ、iOS向けアプリ「Hello Kitty Worl...

○ 平成24年3月期 決算短信

○ 特別損失の計上に関するお知らせ

○ 「シュフモ」登録会員数150万人を突破、「2012年 主婦の全国節電調査（冬季...

ニフティとなら、きっとかなう。  
With Us, You Can.

# ニフティ株式会社

HOME

NIFTY

ニュースリリース

アット・ニフティ

楽しいサービスがいっぱい

@nifty

アクセスマップ

大森から西新宿へ移転いたしました

@nifty Web募金

東日本大震災復興支援  
募金受付中



2012年4月25日 IR 特別損失の計上に関するお知らせ

2012年4月25日 IR 剰余金の配当に関するお知らせ

2012年4月25日 IR 平成24年3月期決算短信

2012年4月11日 IR ニフティ、「@nifty EMOBILE LTE 定額にねんプラン」の提供を開始

2012年4月11日 IR ニフティとサンリオウエーブ、iOS向けアプリ「Hello Kitty World」を台湾で提供開始

2012年4月10日 お知らせ 「@nifty温泉」で「母の日 全国一斉！100のありがとう風呂」特設サイト公開



講師

# 五十嵐邦明

株式会社万葉 エンジニア



「北の空の玄関」新千歳空港直結。  
この便利さがエアターミナル ホテル。

エアターミナル ホテル  
Air Terminal Hotel

Tel. 0123-45-6677 FAX 0123-45-6677

朝食無料サービス  
新千歳空港直結 万葉の湯併設セット入館料無料!

# 連絡

**11月1日の講義は休講です。**





# おみくじアプリ

<http://www.flickr.com/photos/pouchin/7988561427/>

# おみくじアプリ完成予想イメージ



# おみくじアプリづくり

## 手順

1. Rubyでおみくじクラスを作る
2. Railsでアプリを作る
3. アプリにおみくじクラスを組み込む

# おみくじアプリづくり

## 手順

1. Rubyでおみくじクラスを作る
2. Railsでアプリを作る
3. アプリにおみくじクラスを組み込む



# Rubyでおみくじクラスを作る

## 要件

(※ 要件=満たすべき条件)

# Rubyで作るおみくじクラス要件

1. おみくじ、ラッキーカラー、ラッキーな方角を取得可能。内容は以下。

▶ おみくじ

▶ 大吉 / 吉 / 中吉 / 小吉 / 凶

▶ ラッキーカラー

▶ 赤 / 青 / 黄 / 緑

▶ ラッキーな方角

▶ 東 / 西 / 南 / 北

# Rubyで作るおみくじクラス要件

**2. おみくじはランダムで結果が変わるようにしてください。**

**(毎回同じ結果にならないようにしてください。)**

# Rubyで作るおみくじクラス要件

3. irbで次のように実行すると、内容の入ったおみくじの結果が「おみくじオブジェクト」として取得できるようにします。

```
require './omikuji'  
omikuji = Omikuji.new
```

(つまり、omikuji.rbを作っってその中にOmikuji クラスを実装します。)



# Rubyで作るおみくじクラス要件

4. 「おみくじオブジェクト」からは  
次のようにして内容を参照できるようにします。

```
omikuji = Omikuji.new  
omikuji.name # => '大吉'  
omikuji.lucky_color # => '青'  
omikuji.lucky_direction # => '南'
```

# 演習：Rubyで作るおみくじクラス

前述の要件で実装してください。

## ヒント

- ▶ 乱数を得るには `rand` を使います。たとえば `rand(4)` は 0 ~ 3 のどれかを返します。
- ▶ クラス.newしたときには、`initialize` メソッドが呼ばれます。
- ▶ Rubyの文法は講義資料第15回、または「たのしいRuby」、ネットを参照してみてください。

<https://github.com/hitotsubashi-ruby/lecture2012>

# Rubyで作るおみくじクラス動作確認

以下のように動けばOKです。irbで実行してみてください。

```
$ irb
```

```
> require './omikuji'
```

```
> omikuji = Omikuji.new
```

```
> omikuji.name # => '大吉'
```

```
> omikuji.lucky_color # => '青'
```

```
> omikuji.lucky_direction # => '南'
```

**解答例を次のページに書いています。**



# 演習：Rubyで作るおみくじクラス

## 解答例

**omikuji.rb**

```
# coding: utf-8
class Omikuji
  attr_reader :name, :lucky_color, :lucky_direction
  def initialize
    @name = ['大吉', '吉', '中吉', '小吉', '凶'][rand(5)]
    @lucky_color = ['赤', '青', '黄', '緑'][rand(4)]
    @lucky_direction = ['東', '西', '南', '北'][rand(4)]
  end
end
```

# 演習解説：Rubyで作るおみくじクラス

```
# coding: utf-8
class Omikuji
  attr_reader :name, :lucky_color, :lucky_direction
  def initialize
    @name = ['大吉', '吉', '中吉', '小吉', '凶'][rand(5)]
    @lucky_color = ['赤', '青', '黄', '緑'][rand(4)]
    @lucky_direction = ['東', '西', '南', '北'][rand(4)]
  end
end
```

**Omikuji.new** すると **initialize**メソッドが呼ばれ、**@name**, **@lucky\_color**, **@lucky\_direction**が作られます。名前が**@**はじまりの変数はインスタンス変数です。

# 演習解説：Rubyで作るおみくじクラス

```
# coding: utf-8
class Omikuji
  attr_reader :name, :lucky_color, :lucky_direction
  def initialize
    @name = ['大吉', '吉', '中吉', '小吉', '凶'][rand(5)]
    @lucky_color = ['赤', '青', '黄', '緑'][rand(4)]
    @lucky_direction = ['東', '西', '南', '北'][rand(4)]
  end
end
```

**@name = ['大吉', '吉', '中吉', '小吉', '凶'][rand(5)]**

のrand(5)は0~4の値をランダムに返します。 ['大吉', '吉', '中吉', '小吉', '凶']は配列です。配列[x]でx番目の要素を取り出せます。0はじまりなことに注意。例えば ['大吉', '吉', '中吉', '小吉', '凶'][2]は'中吉'です。

# 演習解説：Rubyで作るおみくじクラス

```
# coding: utf-8
class Omikuji
  attr_reader :name, :lucky_color, :lucky_direction
  def initialize
    @name = ['大吉', '吉', '中吉', '小吉', '凶'][rand(5)]
    @lucky_color = ['赤', '青', '黄', '緑'][rand(4)]
    @lucky_direction = ['東', '西', '南', '北'][rand(4)]
  end
end
```

**attr\_reader :name** は **@name**を返すメソッドを作ります。以下と同じです。

```
def name
  @name
end
```



# おみくじアプリづくり

## 手順

1. Rubyでおみくじクラスを作る
2. Railsでアプリを作る
3. アプリにおみくじクラスを組み込む

Rubyで動くおみくじクラスができました。ヽ(´▽`)ノ

# おみくじアプリづくり

## 手順

1. Rubyでおみくじクラスを作る
2. Railsでアプリを作る
3. アプリにおみくじクラスを組み込む

つぎはRailsでアプリを作ってみましょう。  
ここからは演習にします。自分で資料を読みながら  
作ってみてください。

# おみくじアプリづくり

今日の運勢を表示するアプリをつくります。

## 1. Railsアプリをつくる

```
$ rails new fortunes
```

## 2. Rails Root へ移動してページのひながたをつくる

```
$ cd fortunes
```

```
$ bundle exec rails g controller today index
```

## 3. Webサーバを起動する

```
$ bundle exec rails s
```

s は server の略です。 ※終了は Ctrl+c (or Ctrl+Pause)

## 4. ブラウザから以下のURLにアクセスする

<http://localhost:3000/today/index>

こんな画面が出ます→

Today#index

Find me in app/views/today/index.html.erb

# おみくじアプリづくり

## 手順

1. Rubyでおみくじクラスを作る
2. Railsでアプリを作る
3. アプリにおみくじクラスを組み込む

そして、ここまでで作った2つを組み合わせます。



## 5. 今日の運勢を表示させる

**omikuji.rb** を **lib/** の下にコピーします。

**lib/omikuji.rb**

というファイルがあるように配置すればOKです。

# 5. 今日の運勢を表示させる

次に、

app/views/today/index.html.erb

```
<h1>Hello#index</h1>
```

```
<p>Find me in app/views/today/index.html.erb</p>
```

となっているところを以下のように修正します。

```
<% require './lib/omikuji.rb' %>
```

```
<% @omikuji = Omikuji.new %>
```

```
<h1>今日の運勢</h1>
```

```
<h2> <%= @omikuji.name %> </h2>
```

```
<p> ラッキーカラー : <%= @omikuji.lucky_color %> </p>
```

```
<p> ラッキーな方角 : <%= @omikuji.lucky_direction %> </p>
```

さきほど同様にブラウザで以下へアクセス

<http://localhost:3000/today/index>

こんな画面が出ればOKです。 →

今日の運勢

小吉

ラッキーカラー：黄

ラッキーな方角：東

# 解説：ViewにRubyのコードを埋め込む

ViewにRubyのコードを埋め込む2つのパーツがあります。  
`<% %>` と `<%= %>` です。

## `<% %>`

例: `<% @omikuji = Omikuji.new %>`

`<% %>` で囲まれたRubyのコードを実行しますが、結果を出力しません。処理だけを行う場合に使います。

## `<%= %>`

例: `<p> ラッキーカラー : <%= @omikuji.lucky_color %> </p>`

`<%= %>` で囲まれたRubyのコードを実行して、結果を出力します。表示したいときに使います。

## 6. Controllerへ処理を移動する

Viewは表示をする役割を受け持つ部分なので、表示部分だけを残して処理はコントローラへ移動します。

以下のファイルを変更して、ブラウザからアクセスしてみてください。

app/controllers/today\_controller.rb

```
require './lib/omikuji.rb'

class HelloController < ApplicationController
  def index
    @omikuji = Omikuji.new
  end
end
```

app/views/today/index.html.erb

```
<% require './lib/omikuji.rb' %>
<% @omikuji = Omikuji.new %>
<h1>今日の運勢</h1>
<h2> <%= @omikuji.name %> </h2>
<p> ラッキーカラー : <%= @omikuji.lucky_color %> </p>
<p> ラッキーな方角 : <%= @omikuji.lucky_direction %> </p>
```

←この2行を削除

※このように「外から見た振る舞いを変えずにコード整理する」ことをリファクタリングと言います。

# 完成です！



# おつかれさまでした



# はやく終わった人は

- **Codecademy のRuby編をやってみる**

Codecademyはオンライン学習サイトです。

実際にブラウザ上で実行して答えあわせをできるのが便利。

<http://www.codecademy.com/tracks/ruby>

- **ミニツク のRuby入門編をやってみる**

ミニツクはオンライン学習サイトです。(日本語)

説明がとても丁寧でわかりやすいです。

<http://www.minituku.net/>



# Introduction to Control Flow

Course written by [Eric Weinstein](#)

[Section](#)[Q&A Forum \(11\)](#)[Glossary](#)[Scratch Pad](#)

## ✓ 1. How It Works

### 2. If

↓ 問題文

Ruby's `if` statement takes an **expression**, which is just a fancy word for something that has a value (like `4`, `true`, or `pants`). If that expression is `true`, Ruby executes the block of code that follows the `if`. If it's not true (that is, `false`), Ruby doesn't execute that block of code: it skips it and goes on to the next thing.

Here's an example of an `if` statement in action:

```
if 1 < 2
  print "I'm getting printed because one
is less than two!"
end
```

Ruby doesn't care about **whitespace** (spaces and blank lines), so the indentation of the print statement isn't *necessary*. However, it's a convention that Rubyists (Ruby enthusiasts) follow, so it's good to get in the habit now. The block of code following an `if` should be indented two spaces.

When you're done with your `if`, you have to tell Ruby by typing `end`.

Write your own `if` statement in the editor. It

ここにコードを書く

できたらRun→

[Run](#)[Reset](#)[Save](#)

You picked a positive integer!

`==> nil`

That's correct! Next Exercise: If

> █

結果が表示される

[Run Exercise](#)[⌘ + Enter](#)[Save Progress](#)[⌘ + S](#)[Reset Exercise](#)[⌘ + R](#)[Show More](#)



# 講義資料置き場

講義資料置き場をつくりました。  
過去の資料がDLできます。

<https://github.com/hitotsubashi-ruby/lecture2012>  
or

<http://bit.ly/ruby-lecture>

# 雑談・質問用facebookグループ

facebookグループを作りました

<https://www.facebook.com/groups/hitotsubashi.rb>

- 加入/非加入は自由です
- 加入/非加入は成績に関係しません
- 参加者一覧は公開されます
- 書き込みは参加者のみ見えます
- 希望者はアクセスして参加申請してください
- 雑談、質問、議論など何でも気にせずどうぞ～
- 質問に答えられる人は答えてあげてください
- 講師陣もお答えします
- 入ったら軽く自己紹介おねがいします