

# Ruby 講義

## 第12回 Gem

五十嵐邦明

twitter : igaiga555

<http://www.facebook.com/kuniaki.igarashi>



2013.7.4 at 一橋大学  
ニフティ株式会社寄附講義  
社会科学における情報技術と  
コンテンツ作成III

# 五十嵐邦明 講師 株式会社万葉



twitter: igaiga555

<https://github.com/igaiga/>

<http://www.facebook.com/kuniaki.igarashi>

# 濱崎 健吾

Teaching Assistant  
fluxflex, inc(米国法人)



twitter: hmsk

<https://github.com/hmsk/>

<http://www.facebook.com/hamachang>

# 先週の復習

# 継承

既に定義されているクラスを拡張して新しいクラスを作ることを継承といいます。

(既にあるたい焼きの型を利用して、少し違う新しいたい焼きの型をつくるようなものです。)

# 継承

たとえばBookクラスとMagazineクラス(雑誌)を作るとします。

```
class Book
  attr_accessor :title, :price
end
```

```
class Magazine
  attr_accessor :title, :price, :number
end
```

別々に定義を書いてもいいのですが、**共通項**もたくさんあります。そんなときは、継承を使うとすっきり書けます。

# 継承

## 継承を使った書き方

```
class Book  
  attr_accessor :title, :price  
end
```

```
class Magazine < Book  
  attr_accessor :number  
end
```

## 継承を使わない書き方

```
class Book  
  attr_accessor :title, :price  
end
```

```
class Magazine  
  attr_accessor :title, :price, :number  
end
```



class Magazine < Book  
と書くことで、Bookクラスを  
継承したMagazineクラスを  
定義できます。

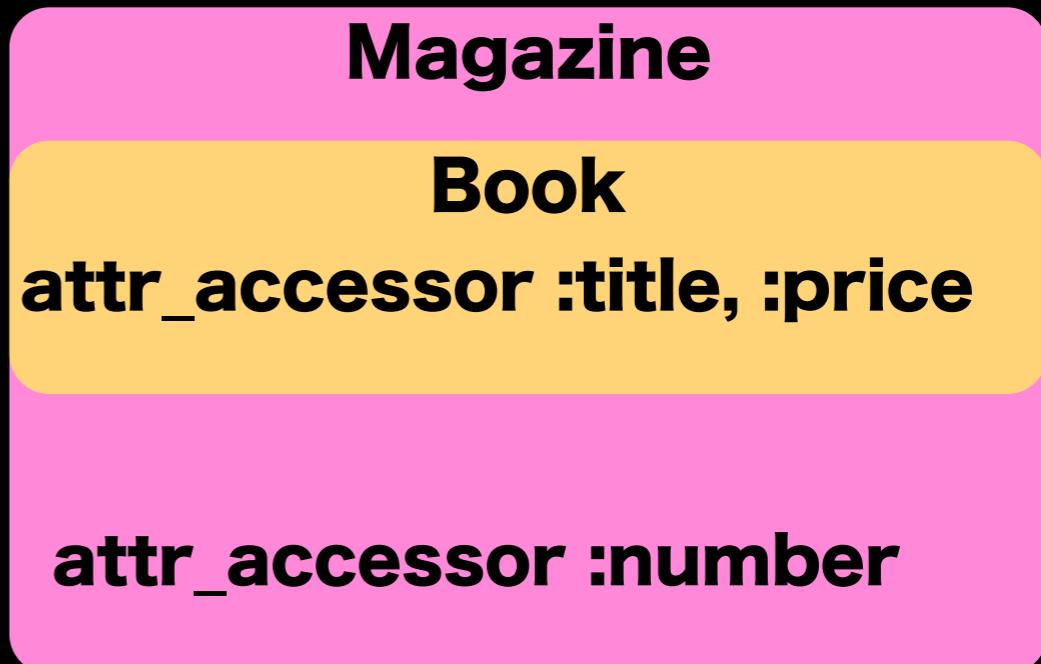
# 継承

## 継承を使った書き方

```
class Book  
  attr_accessor :title, :price  
end
```

```
class Magazine < Book  
  attr_accessor :number  
end
```

## 図解



Bookクラスを継承したMagazineクラスは、Bookクラスの持ち物を受け継ぎます。この例の場合は、Bookクラスの attr\_accessor :title, :price をMagazineクラスでも使えます。加えて、Magazineクラスにある attr\_accessor :number も利用できます。

# 継承

## 継承する場合の書式

```
class クラス名 < スーパークラス名  
  クラスの定義  
end
```

スーパークラスとは、継承元の  
クラス(親クラス)です。

継承したクラスは、親クラスの全てのメソッド、インスタンス変数などを受け継ぎます。

```
class Book  
  attr_accessor :title, :price  
end
```

```
class Magazine < Book  
  attr_accessor :number  
end
```

例えばBookクラスを継承した  
Magazineクラスは、titleとpriceを  
受け継いでいます。例えば、↓のような  
コードを書くことができます。

```
magazine = Magazine.new  
magazine.title = "CanCam"  
p magazine.title #=> "CanCam"
```

# Module

メソッドを共同利用する仕組み

# Module

複数のクラスで同じメソッドを利用したいときにmoduleを使うと重複なく書けるので便利です。

```
module Greeting
  def hello
    puts "Hello!"
  end
end
```

```
class Alice
  include Greeting
end

alice = Alice.new
alice.hello #=> "Hello!"
```

```
class Bob
  include Greeting
end
```

```
bob = Bob.new
bob.hello #=> "Hello!"
```

もしもhelloメソッドで表示する"Hello!"を"こんにちは"に変えたい場合はこのモジュールだけ変更すれば良い

# Moduleの文法

```
module モジュール名  
#メソッド定義  
end
```

```
class Sample  
include モジュール名  
end
```

module 定義

include(読み込み)

# ここから今週の内容

# 目次

Gem

—  
Fragments from  
my work box  
With love from  
Audrey

# Gem

Rubyのライブラリ管理システム

# Gem

ライブラリ：  
他のプログラムから読み込んで利用するためのプログラム

たくさんの便利なライブラリがネット上に公開されています。それらをインストールするなど、管理してくれる仕組みがGemです。

# Gemの例

- Excelファイルを読み書きする
  - twitterなど有名なWebサービスへのアクセスを簡単にしてくれる
  - 画像をリサイズしたりエフェクトをかける
- などなど数万種類のライブラリが公開されています

## gemを紹介、管理しているサイト

**710,437,847  
downloads**

of 40,705 gems cut since July 2009

Welcome to your community RubyGem h

Find your gems easier, publish them faster, and have

do

**LEARN**

**Install RubyGems 1.8.24**

Ruby's premier packaging system

**Browse the Guides**

In depth explanations, tutorials, and references

**Gem Specification**

Your gem's interface to the world

**SHARE**

**`gem update --system`**

Update to the latest RubyGems version

**`gem build foo.gemspec`**

Build your gem

**`gem push foo-1.0.0.gem`**

Deploy your gem instantly

RubyGems.org is the Ruby community's gem hosting service. Instantly publish your gems and install them. Use the API to interact and find out more information about available gems. Become a contributor and enhance the site with your own changes.

**<http://rubygems.org>**

# Gemの使い方

## gemパッケージのインストール

1. shell からgemコマンドを使ってインストール

```
$ gem install gem名
```

2. コードで require "gem名"

```
require "gem名"  
# ここでgem を使うコードを書きます。
```

# Gemの使い方

インストール済のgemパッケージ一覧を表示

shell からgem list コマンドで一覧表示

```
$ gem list
```

# spreadsheet

Excelファイルを操作するgem

# spreadsheet

Excelファイルを読み書きするgem  
Excelがインストールされていなくても、  
Windows以外でも動作します。

shell から以下のコマンドを実行するとインストー  
ルできます。

```
$ gem install spreadsheet
```

# spreadsheet gem 演習

spreadsheet gem をインストールしてから実行してください。

```
$ gem install spreadsheet
```

## a1. Excelファイルを新規作成

```
require "spreadsheet"
```

```
book = Spreadsheet::Workbook.new
sheet = book.create_worksheet
sheet[0,0] = "testing..."
book.write("example.xls")
```

※Spreadsheet::Workbook  
は、 Spreadsheetモジュールの中の  
Workbookクラスの意味です。先週  
説明したモジュールにはこのように種  
類を分けて名前をつける使い方也有  
ります。また次回以降の講義で出てきた  
際に説明します。

生成したexample.xlsを、 Excelか、後述のオープンソースオフィスアプ  
リで開いてみてください。

動作したら、sheet[0,0] の 0,0 の部分を変えてどこに文字列が記入され  
るか見てみてください。

# spreadsheet gem 演習

## a2. 既存のExcelファイルを開いて編集して別名で保存

```
require "spreadsheet"
```

```
book = Spreadsheet.open("example.xls")
sheet = book.worksheet(0)
p str = sheet[0,0]
sheet[1,0] = "hello"
book.write("example2.xls")
```

**example.xls**を開き、画面に内容を表示したり、  
セルに値を追記して別名**example2.xls**として保存するサンプルです。  
Excelファイルのデータを解析するような場合や、  
Excel形式のデータを作る場合に利用できます。

# Office Application

## Libre Office

<http://www.libreoffice.org/download>

## Apache Open Office

<http://www.openoffice.org/ja/download/>

無料で使えるExcelファイルが読み書きできるアプリ  
Win/Mac/Linux 用がそれあります。

# Twitter

twitterを検索したり操作したりするgem

# Twitter

A Ruby interface to the Twitter API.

This project is maintained by [sferik](#)

 Download ZIP

 Download TAR

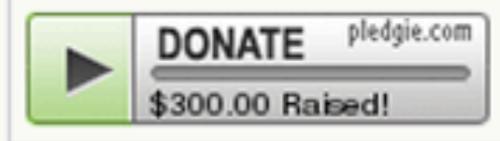
 View On GitHub

# The Twitter Ruby Gem

gem version 4.8.1

build passing

dependencies up-to-date



A Ruby interface to the Twitter API.

## Installation

```
gem install twitter
```

To ensure the code you're installing hasn't been tampered with, it's recommended that you verify the signature. To do this, you need to add my public key as a trusted certificate (you only need to do this once):

<http://sferik.github.io/twitter/>

# twitter gem 演習

twitter gem をインストール

\$ gem install twitter

twitterでユーザーがつぶやいている「最近聞いている曲」を調べる  
nowplaying という言葉入りのtweetを最新10件検索する

```
require "twitter"
```

```
Twitter.configure do |config|
  config.consumer_key = "XXXXXXXXXXXXXXXXXXXX"
  config.consumer_secret = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token_secret = "XXXXXXXXXXXXXXXXXXXX"
end
```

```
Twitter.search("nowplaying", :rpp => 10, :result_type => "recent")[:statuses].each do |status|
  puts "#{status.from_user}:#{status.text}"
end
```

コードを順番に説明していきます。

# twitter 認証キー

最初の Twitter.configure ブロックはtwitter開発者サイトで取得する認証キーを設定します。

```
require "twitter"
```

```
Twitter.configure do |config|
  config.consumer_key = "XXXXXXXXXXXXXXXXXXXX"
  config.consumer_secret = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token_secret = "XXXXXXXXXXXXXXXXXXXX"
end
```

```
Twitter.search("nowplaying", :rpp =>10, :result_type => "recent")[:statuses].each do |status|
  puts "#{status.from_user}:#{status.text}"
end
```

twitter API(情報取得操作窓口)を利用するためには、  
twitterへ登録して許可をもらう必要があります。  
そのための鍵が認証キーです。

# twitter 認証キー 取得

## 以下のtwitter開発者用ページへログインします。

June 11, 2013 Having trouble with your app? API v1 is retired and no longer functional. | Read more →

 Developers

API Health Blog Discussions Documentation

Search



Sign in

Home

## Sign in with your Twitter account

Please log in to access that page.

Username: \*

New to Twitter? [Sign up](#)

Password: \*

[Log in](#)

<https://dev.twitter.com/apps/new>

# twitter 認証キー 取得

Application Details

Name: \*  
lec2013

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description: \*  
Lecture 2013

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website: \*  
<http://example.com>

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.  
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL:

Where should we return after successfully authenticating? For [@Anywhere applications](#), only the domain specified in the callback will be used. [OAuth 1.0a](#) applications should explicitly specify their oauth\_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

アプリ(これから作る  
プログラム)の情報を  
入力します。

\* 印の必須項目を埋め  
ます。 Website欄は  
このアプリの情報を記  
載するページのURL  
を書く欄ですが、ペー  
ジがまだない場合は説  
明文に従って仮で埋め  
ておきます。

# twitter 認証キー 取得

## Developer Rules Of The Road

Last Update: September 5, 2012.

### Rules of the Road

Twitter maintains an open platform that supports the millions of people around the world who are sharing and discovering what's happening now. We want to empower our ecosystem partners to build valuable businesses around the information flowing through Twitter. At the same time, we aim to strike a balance between encouraging interesting development and protecting both Twitter's and users' rights.

So, we've come up with a set of Developer Rules of the Road ("Rules") that describes the policies and philosophy around what type of innovation is permitted with the content and information shared on Twitter.

The Rules will evolve along with our ecosystem as developers continue to innovate and find new, creative ways to use the Twitter API, so please check back periodically to see the current version. Don't do anything prohibited by the Rules and talk to us if you think we should make a change or give you an exception.

If your application will eventually need more than 1 million user tokens, or you expect your [embedded Tweets](#) and [embedded timelines](#) to exceed 10 million daily impressions, you will need to talk to us directly about your access to the Twitter API as you may be subject to additional terms.

Furthermore, applications that attempt to replicate Twitter's core user experience (as described in Section I.5 below) will need our permission to have

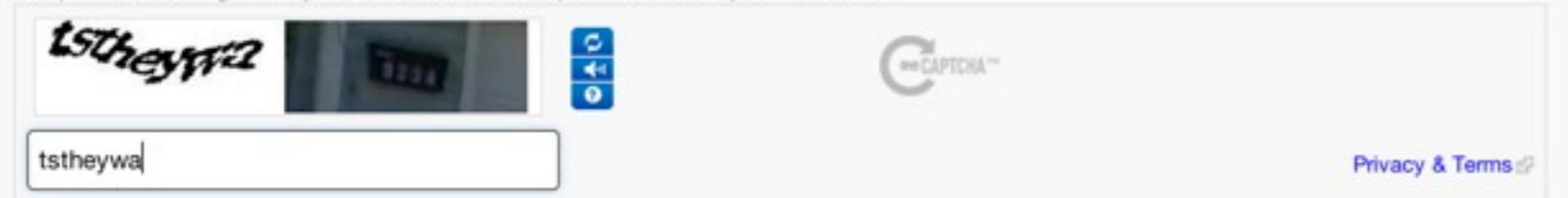
Yes, I agree

By clicking the "I Agree" button, you acknowledge that you have read and understand this agreement and agree to be bound by its terms and conditions.

利用規約に同意して  
チェックします。

### CAPTCHA

This question is for testing whether you are a human visitor and to prevent automated spam submissions.



いたずら防止機能である  
CAPTCHAに文字を入  
力します。(画像に書か  
れている読みづらい文字を  
入力)

全て入力したら "Create your Twitter application" ボタンを押します。

# twitter 認証token 取得

lec2013

Details    Settings    OAuth tool    @Anywhere domains    Reset keys    Delete

 Lecture 2013  
<http://example.com>

**Organization**  
Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

**OAuth settings**  
Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level	Read-only <a href="#">About the application permission model</a>
Consumer key	e1uqYcOjHqgtFB1NRD2l0Q
Consumer secret	1o2t0fd29Ns8WJtvwacHHH0xlznC4mhfQF3erKL44g
Request token URL	<a href="https://api.twitter.com/oauth/request_token">https://api.twitter.com/oauth/request_token</a>
Authorize URL	<a href="https://api.twitter.com/oauth/authorize">https://api.twitter.com/oauth/authorize</a>
Access token URL	<a href="https://api.twitter.com/oauth/access_token">https://api.twitter.com/oauth/access_token</a>
Callback URL	None
Sign in with Twitter	No

**Your access token**  
It looks like you haven't authorized this application for your own Twitter account yet. For your convenience, we give you the opportunity to create your OAuth access token here, so you can start signing your requests right away. The access token generated will reflect your application's current permission level.

[Create my access token](#)

登録すると  
**Consumer key**  
**Consumer secret**  
が得られます。  
メモしておきましょう。

※注意：  
**Consumer secret**は  
秘密にする情報なので  
←こんな風に公開しちゃいけません。

次は  
**Create my access token**  
ボタンを押します。

# twitter 認証token 取得

しばらくするとAccess token, Access token secret  
が表示されます。メモしておきます。

Your access token

Use the access token string as your "oauth\_token" and the access token secret as your "oauth\_token\_secret" to sign requests with your own Twitter account. Do not share your oauth\_token\_secret with anyone.

Access token	4596601-E2E7bNKVy78kDSrGWnlzFhPjPfa55XbXaPR97FPsOY
Access token secret	8aSTcKaksAC4kaP4x9Tz8KBKuCLInH5kEqARoK41YY
Access level	Read-only

[Recreate my access token](#)

※注意：Access token secret は秘密にする情報なので  
こんな風に公開しちゃいけません。

# トークンについて解説

トークン：  
硬貨の代わりに用いられる代用貨幣のこと。

- Wikipedia

# 4つのトークン

公開

公開トークンは  
id のように使う

非公開

非公開トークンは  
password の  
ように使う

**Consumer key**

**Consumer secret**

**Consumer key/secret**

**Access token**

**Access token secret**

**Access token/secret**

ここで言うConsumerはアプリのことです。  
例えば、あるアプリに悪意があることが分かり、  
そのアプリについては認証させないようにする場合  
はこの2つのトークンを無効にすれば遮断できます。

ここで言うAccess tokenは利用  
者1人1人に発行されます。利用者  
がこのアプリに対する認証を禁止す  
る場合には、この2つのトークンを  
無効にします。

# トークンとアプリ、ユーザーの関係



# アプリを利用不可にする場合

利用不可にするアプリの Consumer key/secret を無効化します。  
結果的に、そのアプリを使っているユーザー全員が利用不可になります。

twitter  
アクセス



アプリ

パクドラ

Consumer key



Consumer secret



Consumer key/secret

Consumer key/secret は  
アプリごとに発行される

利用



ユーザー

Access token

Access token secret

Access token/secret

利用



ユーザー

Access token

Access token secret

Access token/secret

Access token/secret は  
ユーザーごとに発行される

# あるユーザーのみアプリを利用不可にする場合

自分の Access token/secret を無効化します。

自分だけはそのアプリがtwitterへアクセスするのを不可能にできます。

twitter  
アクセス



アプリ

パクドラ

Consumer key

Consumer secret

Consumer key/secret

Consumer key/secret は  
アプリごとに発行される



利用



ユーザー

Access token

Access token secret

Access token/secret



利用



ユーザー

Access token

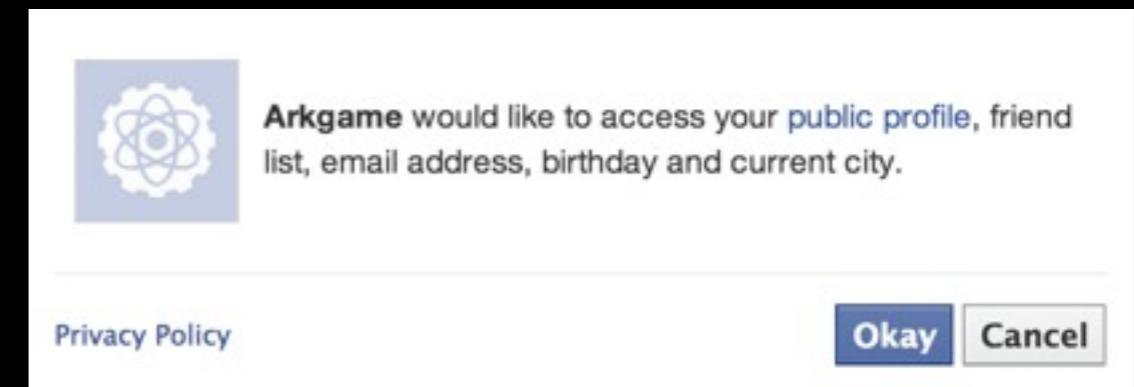
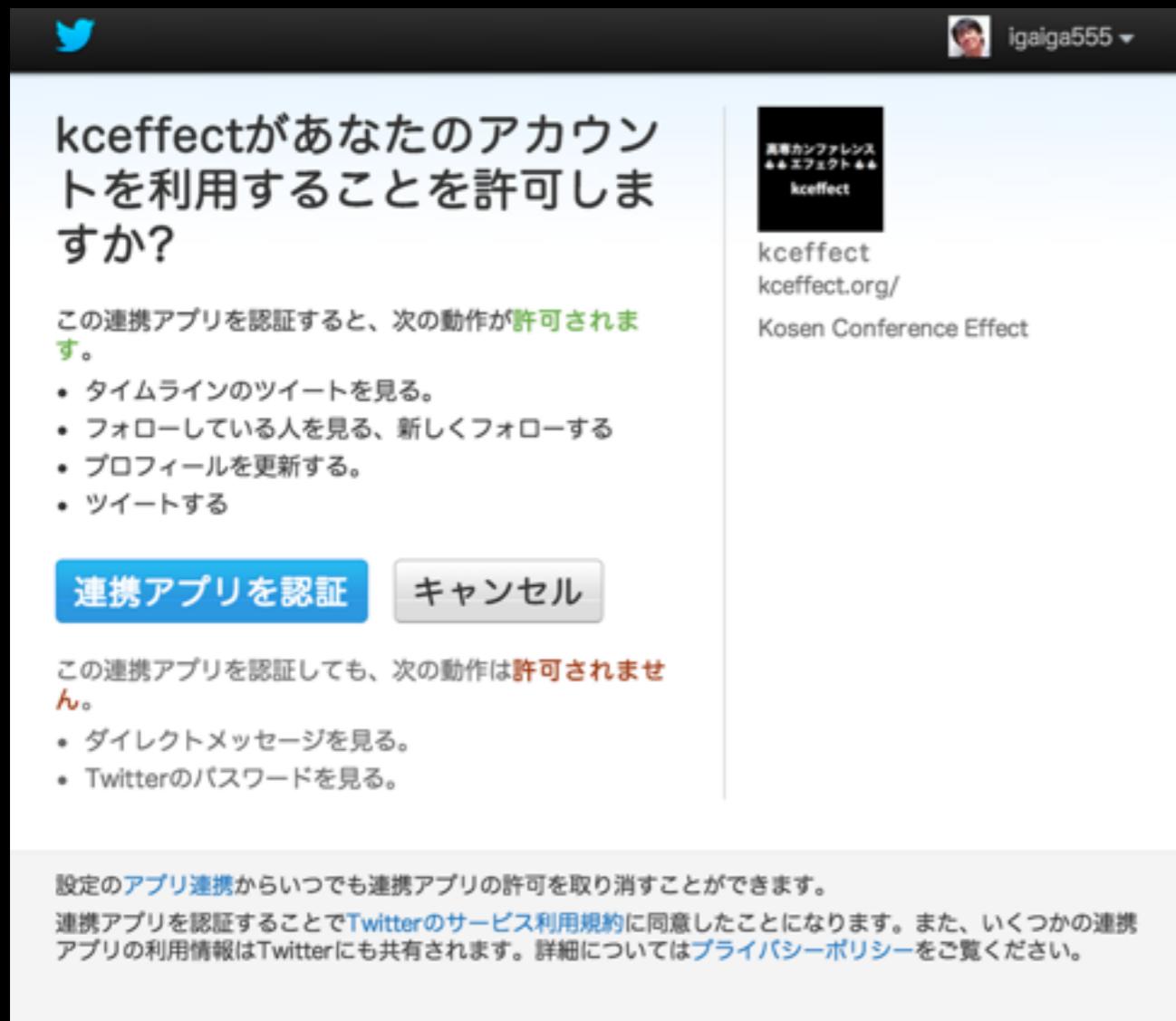
Access token secret

Access token/secret

Access token/secret は  
ユーザーごとに発行される

# twitter, facebook のアプリ認証

twitterやfacebookでこんな画面を見たことありませんか？



**facebook**

**twitter**

これを認証するときに発行されているのも  
このAccess token/secret です。

**Access token**

**Access token secret**

**Access token/secret**

では、コードに  
戻りましょう

# twitter 認証キー

twitterサイトから取得した4つの値を対応する変数に設定します。

```
require "twitter"
```

```
Twitter.configure do |config|
  config.consumer_key = "XXXXXXXXXXXXXXXXXXXX"
  config.consumer_secret = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token_secret = "XXXXXXXXXXXXXXXXXXXX"
end
```

```
Twitter.search("nowplaying", :rpp =>10, :result_type => "recent")[:statuses].each do |status|
  puts "#{status.from_user}:#{status.text}"
end
```

# twitter 認証キー

次は後半部分を説明します。

```
require "twitter"
```

```
Twitter.configure do |config|
  config.consumer_key = "XXXXXXXXXXXXXXXXXXXX"
  config.consumer_secret = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token_secret = "XXXXXXXXXXXXXXXXXXXX"
end
```

```
Twitter.search("nowplaying", :rpp => 10, :result_type => "recent")[:statuses].each do |status|
  puts "#{status.from_user}:#{status.text}"
end
```

# twitter情報取得

twitterでユーザーがつぶやいている「最近聞いている曲」を調べる  
nowplaying という言葉入りのtweetを最新10件検索する

```
この言葉を含む 件数  
Twitter.search("nowplaying", :rpp => 10,  
  :result_type => "recent").each do |status|  
  puts "#{status.from_user}: #{status.text}"  
end      つぶやいたユーザー  つぶやき内容
```

Twitter.searchメソッドを使って検索します。  
引数に検索したい内容を渡します。  
10件取得したつぶやきは、1件ずつブロックが実行され、  
ブロック内変数 status に代入されます。

# 演習解説 文字列表示

```
puts "#{status.from_user}: #{status.text}"
```

"#{変数名}" と書くと、文字列に変数の内容を埋め込むことができます。

```
name = "igarashi"
```

```
puts "Author #{name}" #=> Author igarashi
```

変数nameの中身が表示される

以下の2つのコードは同じ動作です。

```
name = "igarashi"
```

```
text = "Author #{name}"
```



```
text = "Author igarashi"
```

同じ動作

# 演習解説 文字列表示

Rubyには2種類の文字列の表現方法があります。

ダブルクオート（"）とシングルクオート（'）です。

前のページで説明した変数の中身を表示する場合はダブルクオートを使う必要があります。

ダブルクオート："#{変数名}" と書くと変数の中身を表示

```
name = "igarashi"
```

```
puts "Author #{name}" #=> Author igarashi
```

変数nameの中身が表示される

シングルクオート：'#{変数名}' と書くとそのまま表示

```
name = "igarashi"
```

```
puts 'Author #{name}' #=> Author #{name}
```

変数nameの中身が表示されず、そのまま出力される

# 演習解説 引数のHashの省略形

この部分、見慣れない書き方をしています。

```
Twitter.search("nowplaying", :rpp => 10, :result_type => "recent")
```

省略しないで書くと以下のようになります。

```
Twitter.search("nowplaying", {:rpp => 10, :result_type => "recent"})
```

1番目の引数は文字列    2番目の引数はHashで[]を省略

Rubyでよく使われるテクニックで、たとえば複数のオプションを引数として渡したいときにHashを渡すことがあります。キーが説明文の役目をしてくれて読みやすいからです。

その際に曖昧にならなければHashの[]を省略して書けます。

# twitter gem 演習

では実行してみましょう！

```
require "twitter"
```

```
Twitter.configure do |config|
  config.consumer_key = "XXXXXXXXXXXXXXXXXXXX"
  config.consumer_secret = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token_secret = "XXXXXXXXXXXXXXXXXXXX"
end
```

```
Twitter.search("nowplaying", :rpp => 10, :result_type => "recent")[:statuses].each do |status|
  puts "#{status.from_user}:#{status.text}"
end
```

※Windowsで以下のようなエラーが出る場合は次ページ参照

```
C:/RailsInstaller/Ruby1.9.3/lib/ruby/1.9.1/net/http.rb:799:in
`connect': SSL_connect returned=1 errno=0 state=SSLv3 read server
certificate B: certificate verify failed (Twitter::Error::ClientError)
```

# Windows で以下のようなエラーが出る場合の対応方法

```
C:/RailsInstaller/Ruby1.9.3/lib/ruby/1.9.1/net/http.rb:799:in `connect':  
SSL_connect returned=1 errno=0 state=SSLv3 read server certificate  
B: certificate verify failed (Twitter::Error::ClientError)
```

Rubyが使うSSL証明書(セキュリティのために必要)の期限が切れたために発生する問題です。

1. 以下から証明書ファイルをDLして

C:/RailsInstaller/cacert.pem という名前で保存します。

<http://curl.haxx.se/ca/cacert.pem>

2. 以下のコマンドを打ち、環境変数 SSL\_CERT\_FILE にそのパスを設定

```
$ setx SSL_CERT_FILE "C:/RailsInstaller/cacert.pem"
```

3. 開いているコマンドプロンプトは一度閉じて開きなおす

これでエラーが出なくなります。

# twitter gem 演習

```
require "twitter"
```

```
Twitter.configure do |config|
  config.consumer_key = "XXXXXXXXXXXXXXXXXXXX"
  config.consumer_secret = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token = "XXXXXXXXXXXXXXXXXXXX"
  config.oauth_token_secret = "XXXXXXXXXXXXXXXXXXXX"
end
```

```
Twitter.search("nowplaying", :rpp => 10, :result_type => "recent")[:statuses].each do |status|
  puts "#{status.from_user}:#{status.text}"
end
```

検索語"nowplaying"の部分をいろいろ変えて実行してみてください。  
(twitter API は単位時間内に実行できる回数が限られているので、実行しすぎると1時間程度実行できなくなります。)

日本語にする場合はマジックコメント(# coding:utf-8)を忘れずに。  
また、Windowsで結果が文字化けする場合は、以下のようにファイル  
result.txt へ書き出してエディタで開く、を試してみてください。

\$ ruby twitter.rb > result.txt

# Twitter

A Ruby interface to the Twitter API.

This project is maintained by [sferik](#)

 Download ZIP

 Download TAR

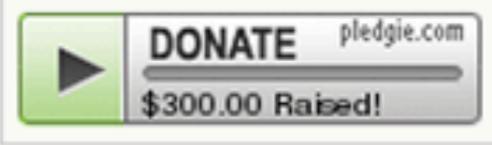
 View On GitHub

## The Twitter Ruby Gem

gem version 4.8.1

build passing

dependencies up-to-date



A Ruby interface to the Twitter API.

**twitter gem 演習上級問題**  
このページの "Usage Examples" 欄などを参考にして  
tweet を投稿するなど、  
いろいろ試してみてください。

Installation

```
gem install twitter
```

To ensure the code you're installing hasn't been tampered with, it's recommended that you verify the signature. To do this, you need to add my public key as a trusted certificate (you only need to do this once):

<http://sferik.github.io/twitter/>



# 參考資料

# 書式

Rubyコード

`puts "abc"`

実行結果

"abc"

実行結果

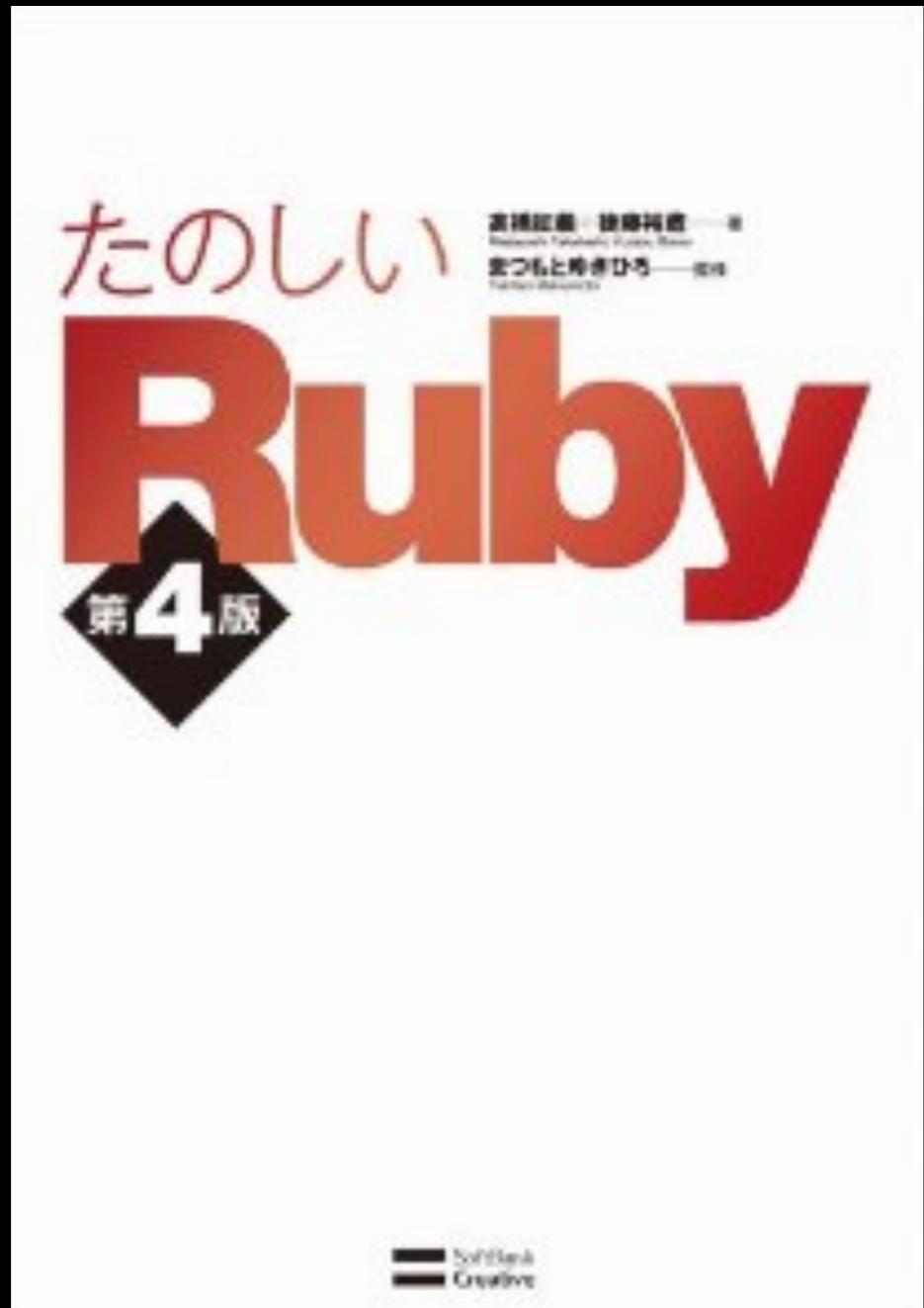
文中では `#=>` で書きます

`p 1+2 #=> 3`

shellコマンド

`$ ls`

# 教科書：たのしいRuby



<http://www.amazon.co.jp/dp/4797372273/>



お買い求めは  
大学生協または  
ジュンク堂池袋店で

# 課題チェック用の付箋の書き方

上の方に学籍番号と名前を書いてください

学籍番号

名前

※この辺に講師陣がクリアした課題番号を書いていきます。

# 講義資料置き場

過去の資料がDLできます。

<https://github.com/igaiga/hitotsubashi-ruby-2013>

# 雑談・質問用facebookグループ

<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
- 参加者はスタッフ(講師・TA)と昨年、今年の受講者です
- 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