

# WEB+DBシステム (入門編)



## 第14回 My Twitterサイトの制作

# 今日のテーマ

- 前回制作したMy Twitterをさらに拡充させて、形にしてみます。
- 新しい構造の実装として、自己参照結合による多対多の構造を実装してみます。

# コミュニケーション(再掲)

フォロー情報の登録を考えてみます。

後期に、一連の技術的な発展系も考えてみて、  
本格的な「つぶやき」コミュニケーション構造を作ってみたい  
と思います。

各ユーザに「顔写真」を登録する。

ユーザが、自由に自分のホームをデザインできる。

フォローしている「つぶやき」とリンクを取る。

なるべく、「システムの設計能力を高める」ための題材として、各  
学生自身のオリジナリティのあるシステム設計をどう学ぶ  
か、シラバスを重視しつつ、アレンジを試みてみます。(予告  
編になってしまいました。)

ただ、実現できるかは・・・

# ツイートする。

自分の「つぶやき」を書き込むのは、ChirpテーブルのデータをNew(新規登録)する形で実装できます。

前回は、「ID」の設定がなかったため、前回のままでNewすると、リレーションが貼れないままでした。

# Userに情報を追加

Userに、ニックネーム(Handle名)を追加する。

顔写真の画像は、リレーションでphotosテーブルからリレーションを設定して実装する。

Add**項目名**To**モデル名**

という名称のMigrationを生成させる。

```
rails generate migration AddNickNameToUsers nick_name:string
```

```
rake db:migrate
```

```
kobayashi-ikuo-no-MacBook:chirpy kobayashi$ rails generate migration AddNickName
ToUsers nick_name:string
  invoke  active_record
  create  db/migrate/20120718151834_add_nick_name_to_users.rb
kobayashi-ikuo-no-MacBook:chirpy kobayashi$ rake db:migrate
== AddNickNameToUsers: migrating =====
-- add_column(:users, :nick_name, :string)
   -> 0.0009s
== AddNickNameToUsers: migrated (0.0010s) =====

kobayashi-ikuo-no-MacBook:chirpy kobayashi$ █
```

# User.rb

app/models/user.rb

で、Attr\_accessibleに、:nick\_name  
を追加する。

```
1 class User < ActiveRecord::Base
2   # Include default devise modules. Others available are:
3   # :token_authenticatable, :confirmable,
4   # :lockable, :timeoutable and :omniauthable
5   devise :database_authenticatable, :registerable,
6         :recoverable, :rememberable, :trackable, :validatable
7   has_many :chirps
8
9   # Setup accessible (or protected) attributes for your model
10  attr_accessible :email, :password, :password_confirmation, :remember_me, :nick_name
11  # attr_accessible :title, :body
12 end
13
```

# User新規登録の修正

app/views/devise/registrations/new.html.erb

を修正し、新規登録の際にnick\_nameを登録するようにする。

```
1 <h2>Sign up</h2>
2
3 <%= form_for(resource, :as => resource_name, :url => registrati
4   <%= devise_error_messages! %>
5
6 <div><%= f.label :email %><br />
7   <%= f.email_field :email %></div>
8
9 <div><%= f.label :nick_name %><br />
10  <%= f.text_field :nick_name %></div>
11
12 <div><%= f.label :password %><br />
13   <%= f.password_field :password %></div>
14
15 <div><%= f.label :password_confirmation %><br />
16   <%= f.password_field :password_confirmation %></div>
17
```

# User編集の登録

app/views/devise/registrations/edit.html.erb  
を修正し、nick\_nameを編集できるようにする。

```
1 <h2>Edit <%= resource_name.to_s.humanize %></h2>
2
3 <%= form_for(resource, :as => resource_name, :url => r
4   <%= devise_error_messages! %>
5
6 <div><%= f.label :email %><br />
7   <%= f.email_field :email %></div>
8
9 <div><%= f.label :nick_name %><br />
10  <%= f.text_field :nick_name %></div>
11
12 <div><%= f.label :password %> <i>(leave blank if you
13   <%= f.password_field :password, :autocomplete => "of
14
15 <div><%= f.label :password_confirmation %><br />
16   <%= f.password_field :password_confirmation %></div>
17
```



# ユーザ情報編集へのリンク

app/views/welcome/index.html.erb

に、ユーザ情報編集へのリンクを追加する。

```
<p><%= link_to 'ユーザ情報編集', [ :edit, :user_registration ] %>
```

```
1 <h1>Chirpy: 私のツイッターによろこそ</h1>
2
3 <p><%= link_to 'ログイン', [ :new, :user_session ] %></p>
4 <p><%= link_to 'ユーザの登録', [ :new, :user_registration ] %></p>
5 <p><%= link_to 'ユーザ情報編集', [ :edit, :user_registration ] %>
6 <p><%= link_to 'パスワードの再発行', [ :new, :user_password ] %></p>
7
```

# テストラン

ここまでの修正で、ユーザ編集で、ニックネームが登録できるか確認する。

**Chirpy: 私のツイッターにようこそ**

[ログイン](#)

[ユーザの登録](#)

**ユーザ情報編集**

[パスワードの再発行](#)

You need to sign in or sign up before continuing.

**Sign in**

Email

Password

Remember me

[Sign up](#)

[Forgot your password?](#)

Signed in successfully.

**Edit User**

Email

Nick name

Password (leave blank if you don't want to change it)

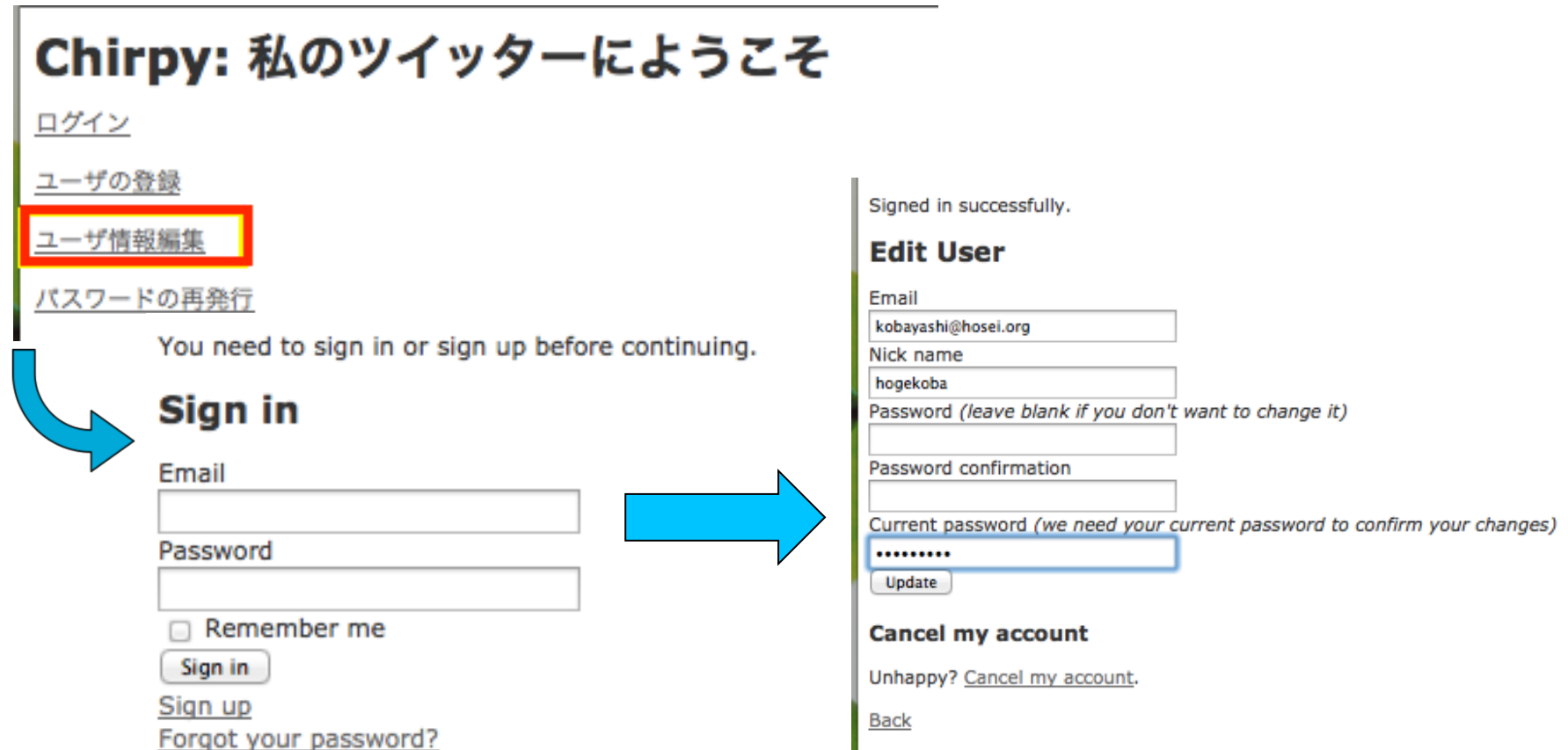
Password confirmation

Current password (we need your current password to confirm your changes)

**Cancel my account**

Unhappy? [Cancel my account.](#)

[Back](#)



# テストラン(2)

新規登録で、ニックネームが登録できるか確認

## Chirpy: 私のツイッターによろこそ

[ログイン](#)

[ユーザの登録](#)

[ユーザ情報編集](#)

[パスワードの再発行](#)



### Sign up

Email

Nick name

Password

Password confirmation

[Sign in](#)

[Forgot your password?](#)

# つぶやきのクリア

「つぶやき」に、ログインユーザ情報を追加します。

前回までの修正でつぶやいたデータは、userへのリレーション情報が保存されていないため、一旦全部削除して下さい。

# chirps\_controller.rb

app/controllers/chirps\_controller.rb

を修正し、createメソッドにuser\_idの設定を1行加える。

```
@chirp.user_id = current_user.id
```

これによって、新規登録されるつぶやきに、ログインしたユーザが結びつけられる。

```
40  
41 # POST /chirps  
42 # POST /chirps.json  
43 def create  
44   @chirp = Chirp.new(params[:chirp])  
45   @chirp.user_id = current_user.id  
46
```

# Viewsからuser\_idを削除

app/views/chirps/\_form.html.erb  
app/views/chirps/show.html.erb  
から、user\_idを扱っている部分を削除する。

```
<li><%= msg %></li>
<% end %>
</ul>
</div>
<% end %>

<div class="field">
  <%= f.label :content %><br />
  <%= f.text_field :content %>
</div>
<div class="field">
  <%= f.label :user_id %><br />
  <%= f.number_field :user_id %>
</div>
<div class="actions">
  <%= f.submit %>
</div>
<% end %>
```

```
show.html.erb
otice %>
<p>
  <b>Content:</b>
  <%= @chirp.content %>
</p>
<b>User:</b>
  <%= @chirp.user_id %>
</p>
<%= link_to 'Edit', edit_chirp %>
<%= link_to 'Back', chirps_path %>
```

# index.html.erb

`app/views/chirps/index.html.erb`  
の、`user_id`の表示を、`user.nick_name`に書き換える。

```
13 <% @chirps.each do |chirp| %>
14 <tr>
15   <td><%= chirp.content %></td>
16   <td><%= chirp.user.nick_name %></td>
17   <td><%= link_to 'Show', chirp %></td>
18   <td><%= link_to 'Edit', edit_chirp_path(chirp) %></td>
19   <td><%= link_to 'Destroy', chirp, :confirm => 'Are you sure?', :me
20 </tr>
21 <% end %>
22 </table>
```

# chirps\_controller.rb

一覧では、まず「自分」がつぶやいた情報だけを表示するようにする。

`app/controllers/chirps_controller.rb`

を修正し、indexメソッドの

`@chirps = Chirp.all`

を

`@chirps = Chirp.find_by_sql("select * from chirps where user_id = #{current_user.id}")`

に書き換える。

`find_by_user_id( current_user.id )`の場合、該当レコードが一つだけだとArray型ではなくChirpクラスが戻るため、エラーになる。

```
1 class ChirpsController < ApplicationController
2   before_filter :authenticate_user!
3   # GET /chirps
4   # GET /chirps.json
5   def index
6     @chirps = Chirp.find_by_sql("select * from chirps where user_id = #{current_user.id}")
7
8     respond_to do |format|
9       format.html # index.html.erb
10      format.json { render :json => @chirps }
11    end
12  end
13 end
```



# テストラン

無事つぶやきが、ニックネームと共に表示され、別のユーザIDで登録したつぶやきが表示されなければ、OK。

Signed in successfully.

## Listing chirps

hogekobaさんのつぶやきです。

**Content User**

[Sign out](#)  
[New Chirp](#)

## Listing chirps

ikuoさんのつぶやきです。

**Content**

**User**

今日が前期最後の授業ですね。

ikuo [Show](#) [Edit](#) [Destroy](#)

でも、レポートを採点しないと・・・

ikuo [Show](#) [Edit](#) [Destroy](#)

[Sign out](#)  
[New Chirp](#)

# 顔画像の登録

画像ファイルの登録の、技術的な説明は、第12回の資料を参照して下さい。

画像用ファイルのクラス名は、Faceとする。

Userテーブルと1対1のリレーションとする。

# Face Class

Face Classのフィールド:

user\_id, integer (UserへのリレーションIndex)

name, string, (ファイル名)

size, integer (画像ファイルサイズ [バイト])

content\_type, string (MIMEタイプ名)

content, blob (content of image file)

今回は、deviseのcontrollerが編集できないため scaffoldで生成することにします。

# Faceクラスの生成

以下のコマンドを1行で入力します。

```
rails generate scaffold Face user_id:integer name:string  
size:integer content_type:string content:binary
```

次に、データベースを生成します。

```
rake db:migrate
```

```
kobayashi-ikuo-no-MacBook:chirpy kobayashi$ rails generate scaffold Face user_id:  
integer name:string size:integer content_type:string content:binary  
  invoke  active_record  
  create  db/migrate/20120718164636_create_faces.rb  
.  
.  
.  
      中略  
.  
  invoke  scss  
  identical  app/assets/stylesheets/scaffolds.css.scss  
kobayashi-ikuo-no-MacBook:chirpy kobayashi$ rake db:migrate  
== CreateFaces: migrating =====  
-- create_table(:faces)  
--> 0.0078s  
== CreateFaces: migrated (0.0079s) =====  
  
kobayashi-ikuo-no-MacBook:chirpy kobayashi$ █
```

# リレーションの設定

faceから見ると、所属するUserは一つだけ、必ず存在する。  
faces.rbに `belongs_to :user` を追加  
user.rbからFaceへのリレーションは`has_one`の関係になる。  
user.rbに `has_one :face` を追加する。  
attribute\_accessibleに、`:face`を追加して下さい。

```
1 class Face < ActiveRecord::Base
2   attr_accessible :content, :content_type, :name, :size, :user_id
3   belongs_to :user
4 end
5
```

```
1 class User < ActiveRecord::Base
2   # Include default devise modules. Others available are:
3   # :token_authenticatable, :confirmable,
4   # :lockable, :timeoutable and :omniauthable
5   devise :database_authenticatable, :registerable,
6         :recoverable, :rememberable, :trackable, :validatable
7   has_many :chirps
8   has_one :face
9
10  # Setup accessible (or protected) attributes for your model
11  attr_accessible :email, :password, :password_confirmation, :remember_me, :nick_name, :face
12  # attr_accessible :title, :body
13 end
14
```

# \_form.html.erbの修正

app/views/faces/\_form.html.erb

を修正する。:multipart => trueの追加。

name, size, content\_type, contentなどの入力フィールドを削除する。

user\_idの表示を削り、

```
<%= f.label current_user.nick_name %><br />
```

```
<div class="field">
```

```
  <%= f.label "顔画像" %><br />
```

```
  <%= file_field :file, :upload %>
```

```
</div>
```

に置き換える。

```
13
14⊖ <div class="field">
15   <%= f.label current_user.nick_name %><br />
16 </div>
17⊖ <div class="field">
18   <%= f.label "顔画像" %><br />
19   <%= file_field :file, :upload %>
20 </div>
21⊖ <div class="actions">
22   <%= f.submit %>
23 </div>
24 <% end %>
```

```
1 <%= form_for @face, :html => { :multipart => true } do |f| %>
2   <% if @face.errors.any? %>
3⊖   <div id="error_explanation">
4     <h2><%= pluralize(@face.errors.count, "error") %> prohibited
5
6⊖   <ul>
```

# faces\_controller.rb

app/controllers/faces\_controller.rb

のdef newメソッドを修正する。

「新規登録」だけではなく、ユーザ画像があったら、選択するようにする。

```
@face = Face.find_by_user_id(current_user.id)
```

```
@face = Face.new if !@face
```

```
24 # GET /faces/new
25 # GET /faces/new.json
26 def new
27   @face = Face.find_by_user_id(current_user.id)
28   @face = Face.new if !@face
29
30   respond_to do |format|
31     format.html # new.html.erb
32     format.json { render :json => @face }
33   end
34 end
```

# faces\_controller.rb

`app/controllers/faces_controller.rb`

のcreateメソッドを修正する。全面的に書き換える。  
(修正途中のまま、時間切れしました。)

動作の切り分けが済んでいません。

顔画像の登録は

<http://127.0.0.1:3000/faces/new>

から行って下さい。

暫定的に、updateメソッドも、createと完全に  
同じとすると、動作します。

本来なら、Helperにmoduleを登録すべき。



# faces#create

```
# POST /faces
# POST /faces.json
def create
  if params[:file]
    @file = params[:file][:upload]
    if @file && @file.respond_to?(:original_filename)
      stat = @file.tempfile.stat
      @face = Face.find_by_user_id( current_user.id )
      face_params = {
        :user_id => current_user.id,
        :name => @file.original_filename,
        :size => stat.size,
        :content_type => @file.content_type,
        :content => @file.read
      }
    end
  end
end
```

```
if @face then
  respond_to do |format|
    if @face.update_attributes( face_params )
      format.html { redirect_to @face, :notice => 'Face was
        successfully updated.' }
      format.json { head :no_content }
    else
      format.html { render :action => "edit" }
      format.json { render :json => @face.errors, :status
        => :unprocessable_entity }
    end
  end
end
else
  @face = Face.new( face_params )
  respond_to do |format|
    if @face.save
      format.html { redirect_to @face, :notice => 'Face was
        successfully created.' }
      format.json { render :json => @face, :status
        => :created, :location => @face }
    else
      format.html { render :action => "new" }
      format.json { render :json => @face.errors, :status
        => :unprocessable_entity }
    end
  end
end
end
end
end
end
```

# faces#create

```
41 # POST /faces
42 # POST /faces.json
43 def create
44   if params[:file]
45     @file = params[:file][:upload]
46     if @file && @file.respond_to?(:original_filename)
47       stat = @file.tempfile.stat
48       @face = Face.find_by_user_id( current_user.id )
49       face_params = {
50         :user_id => current_user.id,
51         :name => @file.original_filename,
52         :size => stat.size,
53         :content_type => @file.content_type,
54         :content => @file.read
55       }
56
57       if @face then
58         respond_to do |format|
59           if @face.update_attributes( face_params )
60             format.html { redirect_to @face, :notice => 'Face was successfully updated.' }
61             format.json { head :no_content }
62           else
63             format.html { render :action => "edit" }
64             format.json { render :json => @face.errors, :status => :unprocessable_entity }
65           end
66         end
67       else
68         @face = Face.new( face_params )
69         respond_to do |format|
70           if @face.save
71             format.html { redirect_to @face, :notice => 'Face was successfully created.' }
72             format.json { render :json => @face, :status => :created, :location => @face }
73           else
74             format.html { render :action => "new" }
75             format.json { render :json => @face.errors, :status => :unprocessable_entity }
76           end
77         end
78       end
79     end
80   end
81 end
```

# Show.html.erb

表示をリダイレクトしたfaces/show.html.erbの中身を以下のように書き換える。

```
<p id="notice"><%= notice %></p>
```

```
<p>  
  <b>User:</b>  
  <%= @face.user.nick_name %>  
</p>
```

```
<%= link_to 'Edit', edit_face_path(@face) %> |  
<%= link_to 'Back', faces_path %>
```

```
1 <p id="notice"><%= notice %></p>  
2  
3 <p>  
4   <b>User:</b>  
5   <%= @face.user.nick_name %>  
6 </p>  
7  
8  
9 <%= link_to 'Edit', edit_face_path(@face) %> |  
10 <%= link_to 'Back', faces_path %>  
11
```

# faces\_helper.html.erb

```
module FacesHelper
  def format_column_value(ar, colname)
    if User === ar
      format_user_column_value ar, colname
    elsif Face === ar
      format_face_column_value ar, colname
    end
  end

  def format_user_column_value( user, colname )
    if colname == 'created_at'
      user.created_at.strftime '%Y-%m-%d %H:%M' if user.created_at
    else
      colname
    end
  end

  def format_face_column_value( face, colname )
    if colname == 'content'
      image_tag face.content, face.size, face.name
    else
      face.send( colname )
    end
  end
end
```

```
1 module FacesHelper
2   def format_column_value(ar, colname)
3     if User === ar
4       format_user_column_value ar, colname
5     elsif Face === ar
6       format_face_column_value ar, colname
7     end
8   end
9
10  def format_user_column_value( user, colname )
11    if colname == 'created_at'
12      user.created_at.strftime '%Y-%m-%d %H:%M' if user.created_at
13    else
14      colname
15    end
16  end
17
18  def format_face_column_value( face, colname )
19    if colname == 'content'
20      image_tag face.content, face.size, face.name
21    else
22      face.send( colname )
23    end
24  end
25 end
```

# faces\_controller.rb

**app/controllers/faces\_controller.rb**  
にfileメソッドを追加する。

```
def file
  face = Face.find params[:id]
  filename = (params[:fileext]) ? "#{params[:filename]}.#{params[:fileext]}" :
    params[:filename]
  if filename != face.name
    render :file => File.join( RAILS_ROOT, 'public', '404.html'),
      :status => 404, :layout => true
  else
    send_data face.content,
      :filename => face.name, :type=>face.content_type
  end
end
```

# faces#file

```
def file
  face = Face.find params[:id]
  filename = (params[:fileext]) ? "#{params[:filename]}.#{params[:fileext]}" :
    params[:filename]
  if filename != face.name
    render :file => File.join( RAILS_ROOT, 'public', '404.html'),
      :status => 404, :layout => true
  else
    send_data face.content,
      :filename => face.name, :type=>face.content_type
  end
end
end
```

# Routingの登録

config/routes.rb

に、fileのパスを登録する。

get 'faces/file' => 'faces#file'

```
1 Chirpy::Application.routes.draw do
2
3   get 'faces/file' => 'faces#file'
4   resources :faces
5
6   get 'chirps', :to => 'chirps#index', :as => :user_root
7   resources :chirps
8
9   devise_for :users
10
11   get "welcome/index"
12
```

# Chirt#index

app/views/chirps/index.html.erb

を修正し、つぶやきに画像を追加します。

```
2
3 <%= current_user.nick_name + "さんのつぶやきです。" %>
4 <table>
5   <tr>
6     <th>Face</th>
7     <th>Content</th>
8     <th>User</th>
9     <th></th>
10    <th></th>
11    <th></th>
12  </tr>
13
14 <%= @chirps.each do |chirp| %>
15   <tr>
16     <td>
17       <%= if chirp.user.face %>
18         <%= if chirp.user.face.content_type =~ /^image\/.*?(png|jpeg|gif)$/ %>
19           <%= image_tag url_for({:action => 'file', :controller => 'faces',
20                                :id=> chirp.user.face.id,
21                                :filename => chirp.user.face.name}), :alt => chirp.user.face.name %></td>
22         <%= end %>
23       <%= end %>
24     </td>
25     <td><%= chirp.content %></td>
26     <td><%= chirp.user.nick_name %></td>
```



# テストラン

つぶやきに顔写真が添えられました。

## Listing chirps

hoge kobaさんのつぶやきです。

Face

Content

User



顔が出ると嬉しい。 hoge koba [Show](#) [Edit](#) [Destroy](#)

[Sign out](#)

[New Chirp](#)

# ここまでで、終わりとします。

自己参照結合によって、フォロワーの人たちのぶつやきが、出てくるようにする修正は、見送りました。

(設計上、論理的に、これだとまだ問題があります。複数の人をフォローしていると、Aさんが、Bさんのフォロワーとして、Bさんに対して書き込んだ内容が、Cさんのつぶやきにも表示されてしまう……。但し、今回は主役は自分一人、ということにします。)

また、構造的にもとても模範になれないところで終わってしまっ、申し訳ありません。続きは、各自が挑戦してみてください。

# 前期を終えて・・・

- 駆け足で、Railsの機能をあれこれと試してみました。また、データベースなどの勉強もしてみました。
- 私のサンプルは、必ずしも「正解」とは限りません。「とにかく動かした」という程度ですので、「あ、先生のは間違ってる」と思ったら、ご連絡をいただけるとありがたいです。
- とにかく、色々と作って試してみるのが、プログラミングを覚える最短距離です。
- 後期は、もっと実践的なWEB+データベースを、たくさん作ってみたいと思います。