

WEB+DBシステム (入門編)



第10回(2016年6月16日)

画像のアップロード機能

今日のテーマ

- 課題
- メモ帳に画像フィールドを追加し、画像をアップロードできるようにする。
- 但し、1対多で、メモ帳1に対して添付画像が複数という構成になるため、memosテーブルにはフィールドは追加しない。

エラー対策について

Scaffoldすると、memosにもcategoriesにも、あちこちにindex.html.erbや、_form.html.erbなどのファイルが生成されます。

修正しているファイルが、memos/indexのつもりが、categories/indexだったり、「うっかり」ミスが出ているケースがあります。

過去の memopad, memopad2を参照しながら、作り直したmemopad3を編集している時など、**編集しているファイルが目的のファイルか、しっかり確認してください。**

画像保存用テーブルの追加

- すべての「メモ」に画像を添付するのではなく、必要に応じて「画像」が複数添付できるように、別のテーブルを用意する。
- 名称はAttachmentとする。
- テーブル名は複数形attachments
- Memoテーブルと1対多のリレーションとする。
- 一つのattachmentsは必ずどれかのmemoに属するが、メモから見てattachmentsが存在しなくてもエラーにはしない。(nullを許容する)

Attachment Classの構造

Attachment Classのフィールド:

memo_id, integer (Memo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)

BLOB: Binary Large Object. 画像などのバイナリデータをそのまま保存する際に利用する。

MIMEタイプとは？

Multipurpose Internet Mail Extension

下記のような指定がある。

text/css

image/jpeg

image/png

application/x-internet-signup

他にも多くの種類がある。

<http://www.geocities.co.jp/Hollywood/9752/mime.html>

SQLite3のデータ型

型	内容
NULL	Null値
INTEGER	符号付きの整数値: 1, 2, 3, 4, 6, 8 バイト
REAL	浮動小数点数, i8 バイト
TEXT	文字列データ、UTF-8, UTF-16BE, UTF-16-LE
BLOB	生データ(バイナリ型)

他のSQLツールなどとの互換性のため、あるいは、プログラマの利便性のために、String型などは最終的にSqlite3のText型にマップされる。

Attachmentクラスの生成

以下のコマンドを1行で入力します。

```
rails generate model Attachment  
memo_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
```

```
[root@cisnote memopad]# rails generate model Attachment memo_id:integer name:string size:integer content_type:string content:binary  
  invoke  active_record  
  create  db/migrate/20150611101016_create_attachments.rb  
  create  app/models/attachment.rb  
  invoke  test_unit  
  create  test/models/attachment_test.rb  
  create  test/fixtures/attachments.yml  
[root@cisnote memopad]# rake db:migrate  
== 20150611101016 CreateAttachments: migrating =====  
-- create_table(:attachments)  
   -> 0.0020s  
== 20150611101016 CreateAttachments: migrated (0.0022s) =====  
[root@cisnote memopad]# █
```


Rails/SQLの予約語に注意

フィールド名(変数名)に、ついっっかりrailsの予約語を使ってしまうと、得体の知れないエラーに悩まされることがあります。

(インタープリタは、コマンドか何かと解釈するため、その語によってどんなエラーになるかわからない。)

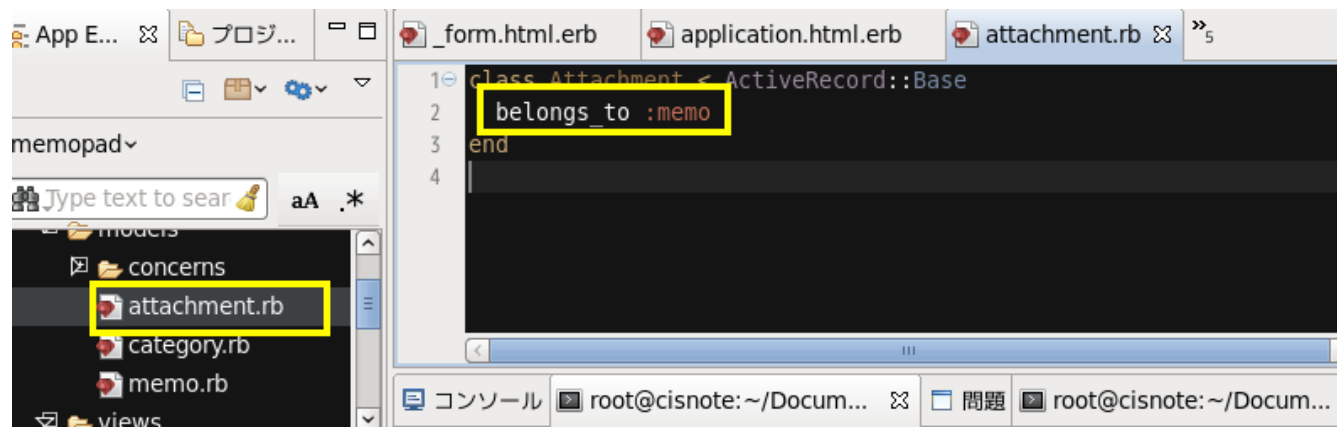
例: content_typeをtypeとして定義すると...

http://www.trail4you.com/TechNote/Ruby/Rails_keyword_list.html

リレーションの設定(1)

attachmentから見ると、所属するメモは一つだけ、必ず存在する。

models/attachment.rbに `belongs_to :memo`を追加



```
1 class Attachment < ActiveRecord::Base
2   belongs_to :memo
3 end
4
```

The screenshot shows a code editor with a file explorer on the left. The file explorer shows a directory structure with folders 'models', 'concerns', and 'views'. Under 'models', there are files 'attachment.rb', 'category.rb', and 'memo.rb'. The 'attachment.rb' file is highlighted. The main editor window shows the content of 'attachment.rb', which is a class definition for 'Attachment' that inherits from 'ActiveRecord::Base'. The code is as follows:

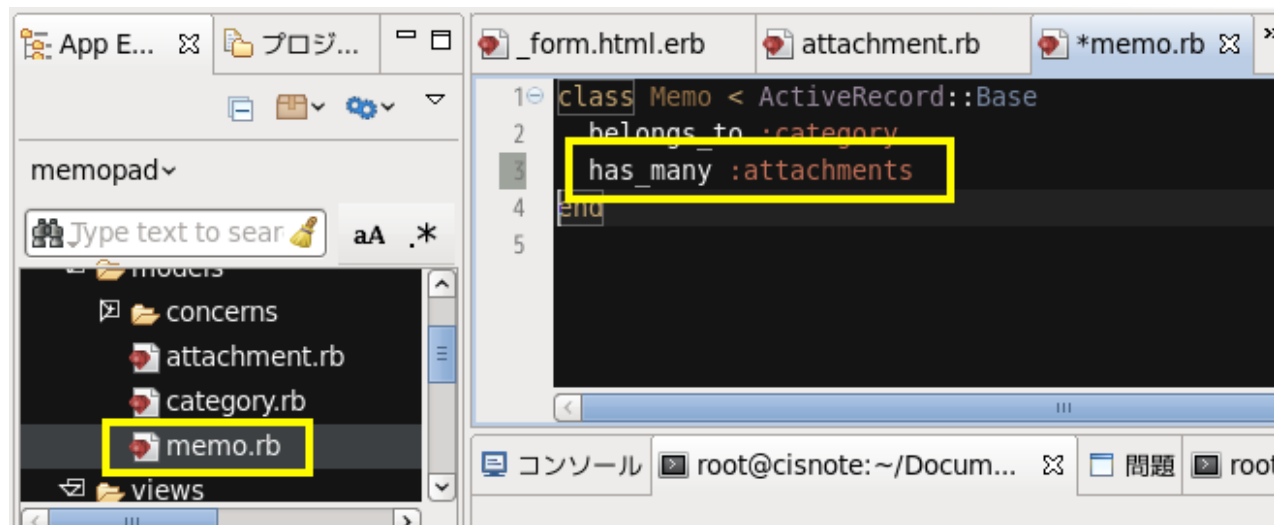
```
1 class Attachment < ActiveRecord::Base
2   belongs_to :memo
3 end
4
```

The line 'belongs_to :memo' is highlighted with a yellow box. The editor also shows tabs for '_form.html.erb', 'application.html.erb', and 'attachment.rb'. The status bar at the bottom shows the terminal output 'root@cisnote: ~/Docum...'.

リレーションの設定(2)

memo.rbからAttachmentへのリレーションはhas_manyの関係になる。

models/memo.rbに `has_many :attachments` を追加する。



memos/_form.html.erbの修正(1)

画像ファイルはサイズが大きいため、uploadは一度にできない。このため、複数のパケットで送信するための指定を行う。(views/memos/_form.html.erb)

```
<%= form_for @memo, :html => { :multipart => true } do |f|  
  %>
```

The image shows a code editor with two windows. The top window displays the code for memos_controller.rb, with lines 1 through 5 visible. The bottom window displays the code for memos/_form.html.erb, with lines 2 through 12 visible. A blue arrow points from the code in the top window to the code in the bottom window, indicating the location of the change. The code in the bottom window shows the form_for tag with the :multipart => true option added, and the error handling code below it.

```
1 <%= form_for(@memo) do |f| %>  
2 <% if @memo.errors.any? %>  
3 <div id="error_explanation">  
4 <h2><%= pluralize(@memo.errors.c  
5  
6  
7  
8  
9  
10  
11  
12
```

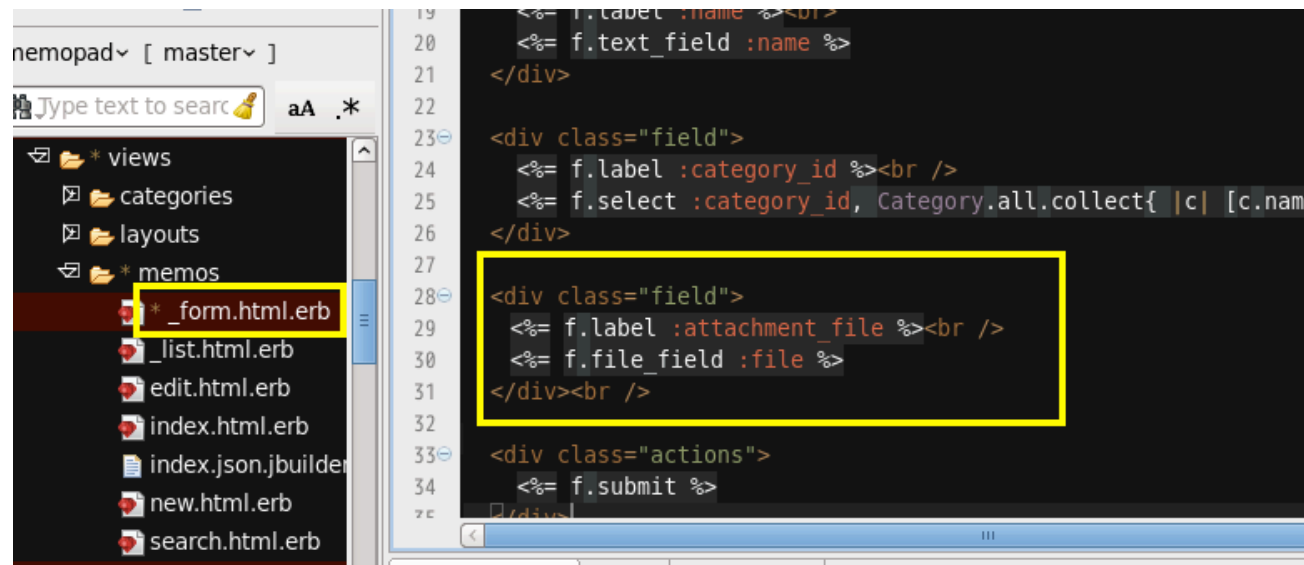
```
1 memos_controller.rb 2 *_form.html.erb 3 show.html.erb 4 routes.rb  
5 form_for @memo, :html => { :multipart => true } do |f| %>  
6 <% if @memo.errors.any? %>  
7 <div id="error_explanation">  
8 <h2><%= pluralize(@memo.errors.count, "error") %> prohibited  
9  
10 <ul>  
11 <% @memo.errors.full_messages.each do |msg| %>  
12 <li><%= msg %></li>  
13 <% end %>  
14 </ul>  
15 </div>  
16 <% end %>
```

_form.html.erbの修正(2)

画像をアップロードするために、_form.html.erb
に以下の修正を加える。

```
<div class="field">  
  <%= f.label :attachment_file %><br />  
  <%= f.file_field :file %>  
</div><br />
```

去年から
変わりました




```
19 <%= f.label :name %><br />  
20 <%= f.text_field :name %>  
21 </div>  
22  
23 <div class="field">  
24   <%= f.label :category_id %><br />  
25   <%= f.select :category_id, Category.all.collect{|c| [c.name,  
26  
27  
28 <div class="field">  
29   <%= f.label :attachment_file %><br />  
30   <%= f.file_field :file %>  
31 </div><br />  
32  
33 <div class="actions">  
34   <%= f.submit %>  
35 </div>
```

memos_controller.rb

createメソッドを修正する。

以下の追加を、saveの下に書き加える。

```
if params[:memo][:file]
  @file = params[:memo][:file]
  @memo.attachments.create :memo_id => @memo.id,
    :name => @file.original_filename,
    :size => @file.size,
    :content_type => @file.content_type,
    :content => @file.read
end
```



去年から
変わりました

createメソッドの全文

```
# POST /memos
# POST /memos.json
def create
  @memo = Memo.new(memo_params)

  respond_to do |format|
    if @memo.save
      if params[:memo][:file]
        @file = params[:memo][:file]
        @memo.attachments.create :memo_id => @memo.id,
          :name => @file.original_filename,
          :size => @file.size,
          :content_type => @file.content_type,
          :content => @file.read
      end
      format.html { redirect_to @memo, notice: 'Memo was successfully created.' }
      format.json { render :show, status: :created, location: @memo }
    else
      format.html { render :new }
      format.json { render json: @memo.errors, status: :unprocessable_entity }
    end
  end
end
```

saveに成功した後に、attachmentをcreateする。理由は、DBに保存しないとmemoのレコードにidが割り振られないため。

createメソッド

```
30
31 # POST /memos
32 # POST /memos.json
33⊖ def create
34   @memo = Memo.new(memo_params)
35
36⊖   respond_to do |format|
37     if @memo.save
38       if params[:memo][:file]
39         @file = params[:memo][:file]
40         @memo.attachments.create :memo_id => @memo.id,
41           :name => @file.original_filename,
42           :size => @file.size,
43           :content_type => @file.content_type,
44           :content => @file.read
45       end
46       format.html { redirect_to @memo, notice: 'Memo was successfully created.' }
47       format.json { render :show, status: :created, location: @memo }
48     else
49       format.html { render :new }
50       format.json { render json: @memo.errors, status: :unprocessable_entity }
51     end
52   end
53 end
54
```


次は、表示用の修正

memos_controller.rbにfileメソッドを追加する。

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

memos_controller.rb

```
memos_controller.rb ✕  _form.html.erb  application.html.erb  attachment.rb  memo.rb  »4
77     format.html { redirect_to memos_url, notice: 'Memo was successfully destroyed.' }
78     format.json { head :no_content }
79     end
80   end
81
82   def file
83     attachment = Attachment.find params[:id]
84     filename = (params[:fileext]) ? "#{params[:filename]}.#{params[:fileext]}" : params[:filename]
85     if filename != attachment.name
86       render :file => File.join( RAILS_ROOT, 'public', '404.html'),
87             :status => 404, :layout => true
88     else
89       send_data attachment.content,
90               :filename => attachment.name, :type=>attachment.content_type
91     end
92   end
93
94   private
95   # Use callbacks to share common setup or constraints between actions.
```

ここでよく起こすエラー

このメソッドを、他のメソッドの中に定義する人
インデントがきちんとしていなくて、「適当」に
追加した結果、他のメソッドの中に入ってしまった
っている場合、エラーになります。

public:ではなく、private:のメソッドとして定義し
てしまった場合、findメソッドがないというエラ
ーになります。

deleteの下、privateの上あたりに追加してく
ださい。

memos_helperの修正

```
module MemosHelper
  def format_column_value(ar, colname)
    if Memo === ar      # 引数arが Memoクラスのインスタンスだったら
      format_memo_column_value ar, colname
    elsif Attachment === ar  # 引数arが Attachmentクラスのインスタンスだったら
      format_attachment_column_value ar, colname
    end
  end
end

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

def format_attachment_column_value( atch, colname )
  if colname == 'content'
    image_tag atch.content, atch.size, atch.name  # 画像の展開は、ここ！
  else
    atch.send( colname )
  end
end
end
```

show.html.erbの修正

```
<p>
  <% if @memo.attachments.length>0 %>
    <b>Attachments</b>
    <table border="1">
      <tr>
        <% for column in @memo.attachments.content_columns %>
          <th><%= column.human_name %></th>
        <% end %>
      </tr>
      <% for attachment in @memo.attachments %>
        <tr>
          <% for column in @memo.attachments.content_columns %>
            <% if column.name == 'content' &&
              attachment.content_type =~ /^image\.(png|jpeg|gif)$/ %>
              <td><%= image_tag url_for({:action => 'file', :id=> attachment.id,
                :filename => attachment.name}), :alt => attachment.name %></td>
            <% else %>
              <td><%= format_column_value( attachment, column.name) %></td>
            <% end %>
          <% end %>
        </tr>
      <% end %>
    </table>
  <% end %>
</p>
```

routes.rbにメソッドを追加する。

config/routes.rbを開き

```
resources :memos
```

よりも上の位置に、

```
get 'memos/file' => 'memos#file'
```

を追加する。

```
1 Rails.application.routes.draw do
2   resources :categories
3
4   get 'memos/search' => 'memos#search'
5   get 'memos/file' => 'memos#file'
6
7   resources :memos
8   # The priority is based upon order of creation: first created -
```

resourcesの下に書いたエラー

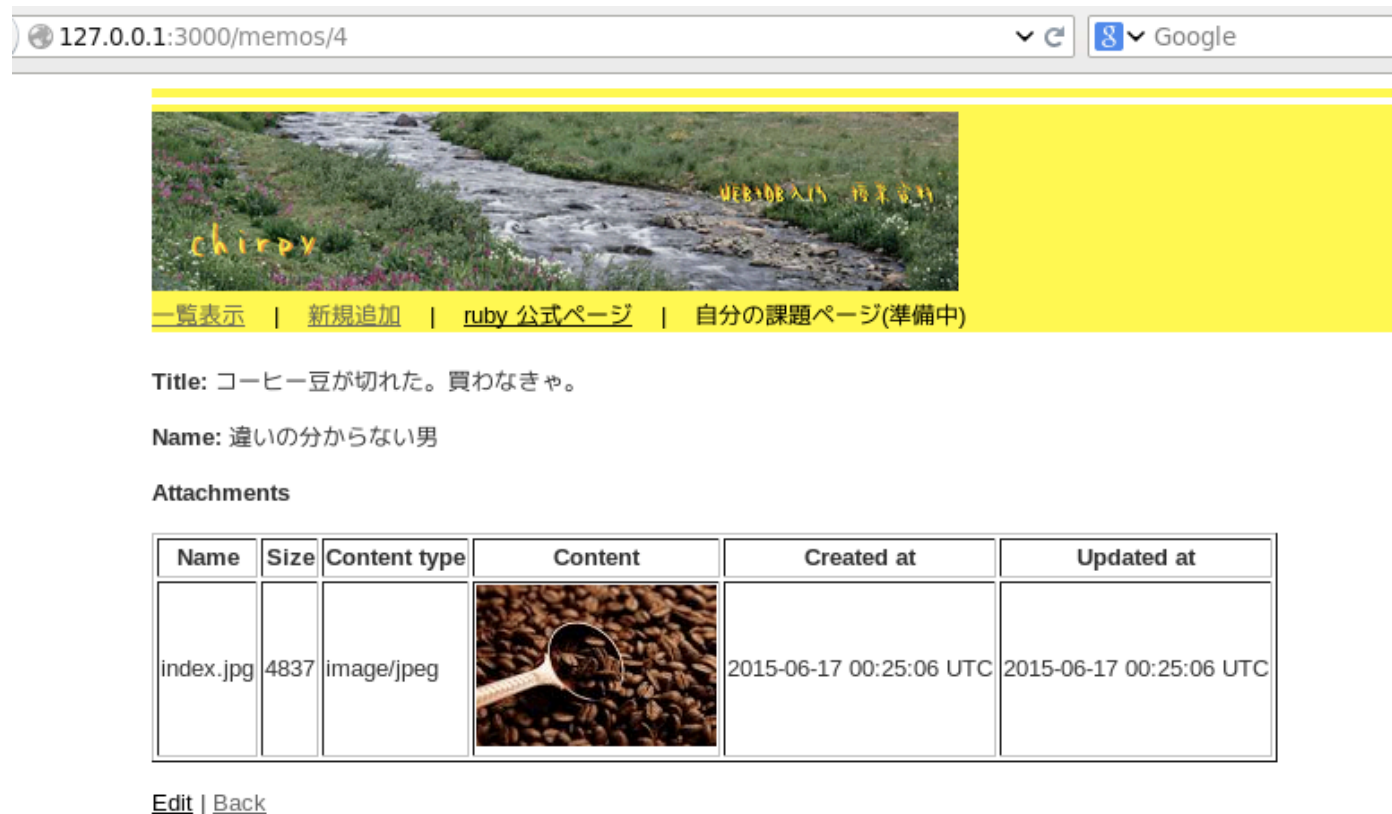
‘file’をidの数字とみなして、showメソッドを呼び出しに行くため、「file」というidのデータが読み出せない」というエラーになります。

Routingテーブルのマッピングに当てはめて、エラーの原因を考えてみてください。

ここまでの修正で

Showの画面は以下ようになります。

127.0.0.1:3000/memos/4 Google




chirpy WEBPAGE入門 構築資料

[一覧表示](#) | [新規追加](#) | [ruby 公式ページ](#) | [自分の課題ページ\(準備中\)](#)

Title: コーヒー豆が切れた。買わなきゃ。

Name: 違いの分からない男

Attachments

Name	Size	Content type	Content	Created at	Updated at
index.jpg	4837	image/jpeg		2015-06-17 00:25:06 UTC	2015-06-17 00:25:06 UTC

[Edit](#) | [Back](#)

Memoの一覧に画像を表示

Indexに画像を表示する修正を行います。

ついでに、罫線枠を設定し、Titleの横幅を200Pixelに制限し、また、画像のサイズは、表示する際に60x60に強制的に設定しました。

```

<% if @memos.size==0 then %>
メモは登録されていません。
<% else %>
<table border="1">
  <tr>
    <th></th>
    <th>Title</th>
    <th>Name</th>
    <th>Category</th>
    <th></th>
    <th></th>
    <th></th>
  </tr>

  <% @memos.each do |memo| %>
    <tr>
      <td>
        <% if memo.attachments.length>0 %>
          <% for attachment in memo.attachments %>
            <% if attachment.content_type =~ /^image\/.*?(png|jpeg|gif)$/ %>
              <%= image_tag url_for({:action => 'file', :id=> attachment.id,
                :filename => attachment.name}), :alt => attachment.name,
                :size => "60x60" %>
            <% end %>
          <% end %>
        <% end %>
      </td>
      <td width="200"><%= memo.title %></td>
      <td><%= memo.name %></td>
      <td><%= memo.category.name %></td>
      <td><%= link_to 'Show', memo %></td>
      <td><%= link_to 'Edit', edit_memo_path(memo) %></td>
      <td><%= link_to 'Destroy', memo, method: :delete, data: { confirm: 'Are you sure?' } %></td>
    </tr>
  <% end %>
</table>
<% end %>

<%= form_tag "/memos/search", :method=>"get" do %>
  <% @options = options_for_select( Category.find(:all).
    collect{|c| [c.name,c.id]}) %>
  <label>分類</label>
  <%= select_tag 'categ_id', @options %>
  <%= submit_tag '選択' %><br />
<% end %>
<br />

<%= link_to 'New Memo', new_memo_path %>

```

_list.html.erb

コピペできますが、Backslashが¥になっただら、入力し直してください。

_list.html.erb


```
memos_controller.rb  _list.html.erb  routes.rb  show.html.erb  memos_helper.rb

7 <thead>
8   <tr>
9     <th></th>
10    <th>Title</th>
11    <th>Name</th>
12    <th>分類</th>
13    <th colspan="3"></th>
14  </tr>
15 </thead>
16
17 <tbody>
18   <%= @memos.each do |memo| %>
19     <tr>
20       <td>
21         <%= if memo.attachments.length>0 %>
22           <%= for attachment in memo.attachments %>
23             <%= if attachment.content_type =~ /^image\/*.*?(png|jpeg|gif)$/ %>
24               <%= image_tag url_for({:action => 'file', :id=> attachment.id,
25                                     :filename => attachment.name}), :alt => attachment.name,
26                                     :size => "60x60" %>
27             <%= end %>
28           <%= end %>
29         <%= end %>
30       </td>
31       <td width="200"><%= memo.title %></td>
32       <td><%= memo.title %></td>
```

新しい、index 画面



私の独り言的メモです

	Title	Name	分類	
	5時限目は疲れる	by教師	バイト	Show Edit Destroy
	5時限目だけど頑張るぞ	by教師	レポート提出	Show Edit Destroy
	コーヒー豆が切れた。買わなきゃ。	違いの分からない男	買い物メモ	Show Edit Destroy

分類

[独り言の追加](#)

メモをdestroyした時の処理

Memoが削除された時に、画像が削除されないと、リレーション先の存在しない画像が呼び出されないまま蓄積されてしまいます。

destroyを実行する前に、以下を実行させます。

memos_controllerのdestroyアクションに5行を追加します。

```
if @memo.attachments.length > 0
  for attachment in @memo.attachments
    attachment.destroy
  end
end
```

Destroy メソッド

```
memos_controller.rb ✕  _list.html.erb  routes.rb  show.html.erb  memos_helper.rb  index.  
69   end  
70   end  
71  
72   # DELETE /memos/1  
73   # DELETE /memos/1.json  
74   def destroy  
75     if @memo.attachments.length > 0  
76       for attachment in @memo.attachments  
77         attachment.destroy  
78       end  
79     end  
80     @memo.destroy  
81     respond_to do |format|  
82       format.html { redirect_to memos_url, notice: 'Memo was successfully destroyed.' }  
83       format.json { head :no_content }  
84     end  
85   end  
86  
87   def file
```

今日の授業を休んだ人は・・・

画像がUploadされている画面のスクリーンショットをつけて、作業内容を報告して下さい。

この提出で、出席に切り換えます。

編集したのに画像が変わらない！

そこは、自分で解決しましょう。

【一つの解決策】

元々あった画像をまず削除する。

新たにuploadしたファイルを登録する。

次回予告

- 第11回は、ログイン機能（登録者のみ閲覧可能）の機能を追加します。
- この段階で、総復習として「MyTwitter」として、ゼロから作り直してみます。