

WEB+DBシステム(入門編)



第4回(2016年4月28日)

HTMLとCSS

今日のテーマ


- HTML(Hyper Text Markup Language)を読む
- CSS(Cascading Style Sheet)の使い方を理解する。
- 画面の修飾、記述方法を調べる。
 - タグについて学ぶ。
- HTML+CSSでWEB画面を記述する方法について実際に試し、理解を深める。

メインの表示画面を見る

- memopad/app/views/layouts/にある application.html.erbを開く

```
memos_controller.rb  _form.html.erb  application.html.erb ✖
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>Memopad</title>
5    <%= stylesheet_link_tag "application", media: "all", "data-turbolinks-track" => true %>
6    <%= javascript_include_tag "application", "data-turbolinks-track" => true %>
7    <%= csrf_meta_tags %>
8  </head>
9  <body>
10
11  <%= yield %>
12
13  </body>
14  </html>
15
```

HTMLのタグ

- `<tag>` 記述 `</tag>`
 - タグと呼ばれるキーワードが並べられる。
 - タグは、必ず対応している必要がある。
 - 単独で動作するタグの場合、`<tag />`と記述すると開いてすぐに閉じる記述となる。
- `<tag1>` `<tag2>` 記述 `</tag2>` `</tag1>`
- 入れ子構造になった場合、内側から閉じていく。

全体構成

```
<html>
```

```
  <!-- コメント -->
```

```
  <!-- 全体が、htmlのタグで囲まれる -->
```

```
  <head>
```

```
    <!-- headタグで囲まれているのがヘッダ部 -->
```

```
  </head>
```

```
  <body>
```

```
    <!-- bodyタグで囲まれている部分に、本体が入る -->
```

```
  </body>
```

```
</html>
```

ブロック構造を常に意識する

C言語やJavaの場合は、{ } で囲まれた領域が一つのブロックになる。

Ruby では、defなどから end までがブロック

閉じるまでが一つの「**ブロック**」となる。
ブロックの開始は、def, module, ifなど

```
1 require File.expand_path('.')
2
3 require 'rails/all'
4
5 if defined?(Bundler)
6   # If you precompile assets
7   Bundler.require(*Rails.groups)
8   # If you want your assets
9   # Bundler.require(:default)
10 end
11
12 module Myshop
13   class Application < Rails
14     # Settings in config/en
```

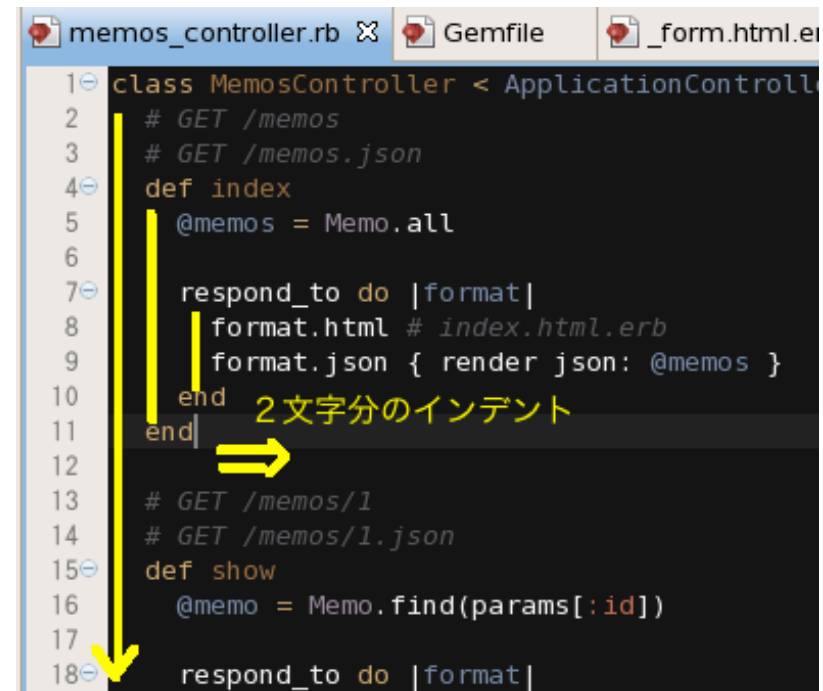
```
57 config.assets.enabled
58
59 # Version of your assets
60 config.assets.version
61 end
62 end
63
```

ソースコードにはインデントを！

プログラムの「段下げ」のことを、**インデント**といいます。
インデントーションがきちんとしていないと、ブロック構造が見やすくなりません。

インデントが書けないプログラマは、同僚から嫌われます！
(しつけがなってない・・・)

空白2文字分で良いでしょう。
インデントは必ずつけて下さい。



```
1 class MemosController < ApplicationController
2   # GET /memos
3   # GET /memos.json
4   def index
5     @memos = Memo.all
6
7     respond_to do |format|
8       format.html # index.html.erb
9       format.json { render json: @memos }
10    end
11  end
12
13  # GET /memos/1
14  # GET /memos/1.json
15  def show
16    @memo = Memo.find(params[:id])
17
18    respond_to do |format|
```

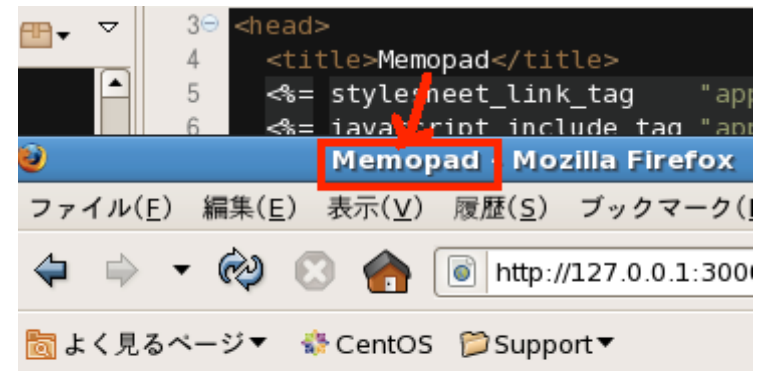
ヘッダ部

□ `<head></head>` で囲まれた部分がヘッダ部

□ ヘッダ部には、`title`タグなどがくる。

□ `<title></title>`

- `title`で指定された文字列が、ブラウザの「見出し」になる。



□ `<meta ····` (次ページ)

□ `<style id="···"></style>`

- 書式を「名前」で指定する記述

マルチフレーム

- `<frame>` `</frame>` で囲まれた部分が、それぞれ独立したhtmlの形態になる。
- `<frameset>` `</frameset>` の中に、複数の `<frame>` `</frame>` ブロックを記述する。
- 現在は、マルチフレームの構造を使っていない。

フレームを左右に2分割する例

```
<html>
<head>
  <title>わたしのブログ</title>
  <META NAME="description" CONTENT="ブログ,〇×日記">
  <META http-equiv="content-type" content="text/html">
</head>
<a name="_top"></a>
<frameset border="0" cols=200,*>
  <frame src="menu.htm" marginwidth=10 scrolling=yes name="menu">
  <frame src="bodyTop.htm" marginheight=20 marginwidth=20
    scrolling="auto" name="main">
</frameset>
</html>
```

歴史的な構造です。忘れていいと思います。
何かの折りに見かけたら、「あれか」と思って下さい。

bodyブロック

□ `<body>` `</body>`タグで囲まれた部分が画面本体の記述になる。

- <http://www.tagindex.com/>

- などを参照のこと

□ ブロック内で使われるタグ（一部抜粋）

- `table / th / tr / td` 「表」の記述、行、列

- `h1 / h2 / h3 ...` 見出し行

- `br / hr` 改行、横線

- `b / i / u / del` 太字など文字修飾

- `a` リンク

- `img` 画像表示

application.html.erbでのbody記述

- `<%= yield =>`
 - とだけ記されている。
 - yield部分でアクションが実行される。
- layouts/application.html.erbは、共通の「型枠」で、型枠の中身としてはめ込む機能はMemos/index.html.erbなどで記載する。
- yieldは、他にも「アクションを実行する」部分で使われる言葉なので、覚えておくとよい

埋め込みruby (Embedded Ruby)

<% %> Ruby言語で解釈されるTAG

<% %>

rubyを主に制御構造に記述する。

<%= %>

rubyで変換した結果をHTMLに出力する。

table記載

- tableは「表」だが、画面全体を升目状に区切って使う際のテクニックとしても利用される。
- 「表現方法」は、文法とは別に自習して下さい。

<table>

「表」の開始

<tr>

横方向「行」の始め

<td>

各列の「データ」の始め

</td>

</tr>

</table>

index.html.erbの中身

```
1 <h1>Listing memos</h1>
2
3 <table>
4   <tr>
5     <th>Title</th>
6     <th>Name</th>
7     <th></th>
8     <th></th>
9     <th></th>
10  </tr>
11
12  <%= @memos.each do |memo| %>
13    <tr>
14      <td><%= memo.title %></td>
15      <td><%= memo.name %></td>
16      <td><%= link_to 'Show', memo %></td>
17      <td><%= link_to 'Edit', edit_memo_path(memo) %></td>
18      <td><%= link_to 'Destroy', memo, method: :delete, data: { confirm: 'Are you
19    </tr>
20  <%= end %>
21 </table>
22
23 <br />
24
25 <%= link_to 'New Memo', new_memo_path %>
```

一覧表的な画面として、
テーブル構造が基本となっている。

tableによるレイアウト編集

- `<table>` ... `</table>`でテーブル全体
 - `<tr>` 行記述 `</tr>`で、行を区切る
 - `<td>` 1桁 `</td>`で、一つの枠を区切る
 - このtd に `colspan`や`rowspan`を組み合わせて、全体の枠を作るテクニックはよく使われる。
 - `align = "center" / "left"`などで、「中央揃え」、「左揃え」などを指定する。
 - `width="200" height="60"`などで、幅、高さを指定

<code><td rowspan="4"></code> 縦に4つ連結	<code><td colspan="3"></code> 横に3つ連結		

見出し行 / 改行 / 横線

- `<h1>`このページのタイトル`</h1>`
- 数字が大きくなるほど見出しレベルが下がってくる。
 - (どんどんと小さくなる。)
- htmlでは、通常の制御文字(改行、タブなど)は意味を持たない。
- 改行する時は、`
`タグを必ず入れる。
- 横線を引く時は、`<hr />`

文字修飾

- `` 太字 `` bold
- `<i>` 斜体字 `</i>` italic
- `<u>` 下線 `</u>` underline
- `` 取り消し線 `` delete
- ``
文字を"2"大きくし、色は赤にする ``

リンク

- ``
- `ファイルへのリンク`
- `ファイル内へのリンク`
- `<a href="http://www.どこかのサイト/"
target="_blank">どこかのサイトへのリンクを新しい
ページで開く`

画像表示

- ``
- 画像タグ
 - `src="ファイル名"`で、表示する画像ファイルを指定
 - `height="高さ"` ピクセルで高さを表示
 - `width = "幅"` ピクセルで幅を表示

- 但し、railsではリソース管理のため
 - `app/assets/images`の下に画像を置き、
 - `<%= image_tag('ファイル名') %>`
 - で記述する。

段落ブロック

- `<div> </div>`
- 囲まれた内部が一つの「段落」として表示される。
- 通常、`align="left"`, `"center"`などのそろえる位置を記したり、さらに「様式」を定義したスタイルシートを引用し、`id=スタイルID`などを記して記載する。

- 他に、段落を区切る機能があるタグ
 - `<p> </p>`,
 - `<blockquote> </blockquote>`

例題演習

- 見出し部分に画像をはめ込んでみる。
- 好きな画像を1枚、jpg、gif、pngなどの形式で保存する。
 - 保存先は、app/assets/images

application.html.erbファイルに、

```
<%= image_tag('banner.gif',  
  :size=> '450x100',  
  :alt => '私の専用メモ帳') %>
```

などと書く。(ファイル名、サイズは各自調整のこと)

- タグを埋め込んで、画像を表示させる。

ホストOSとゲストOSとの共有

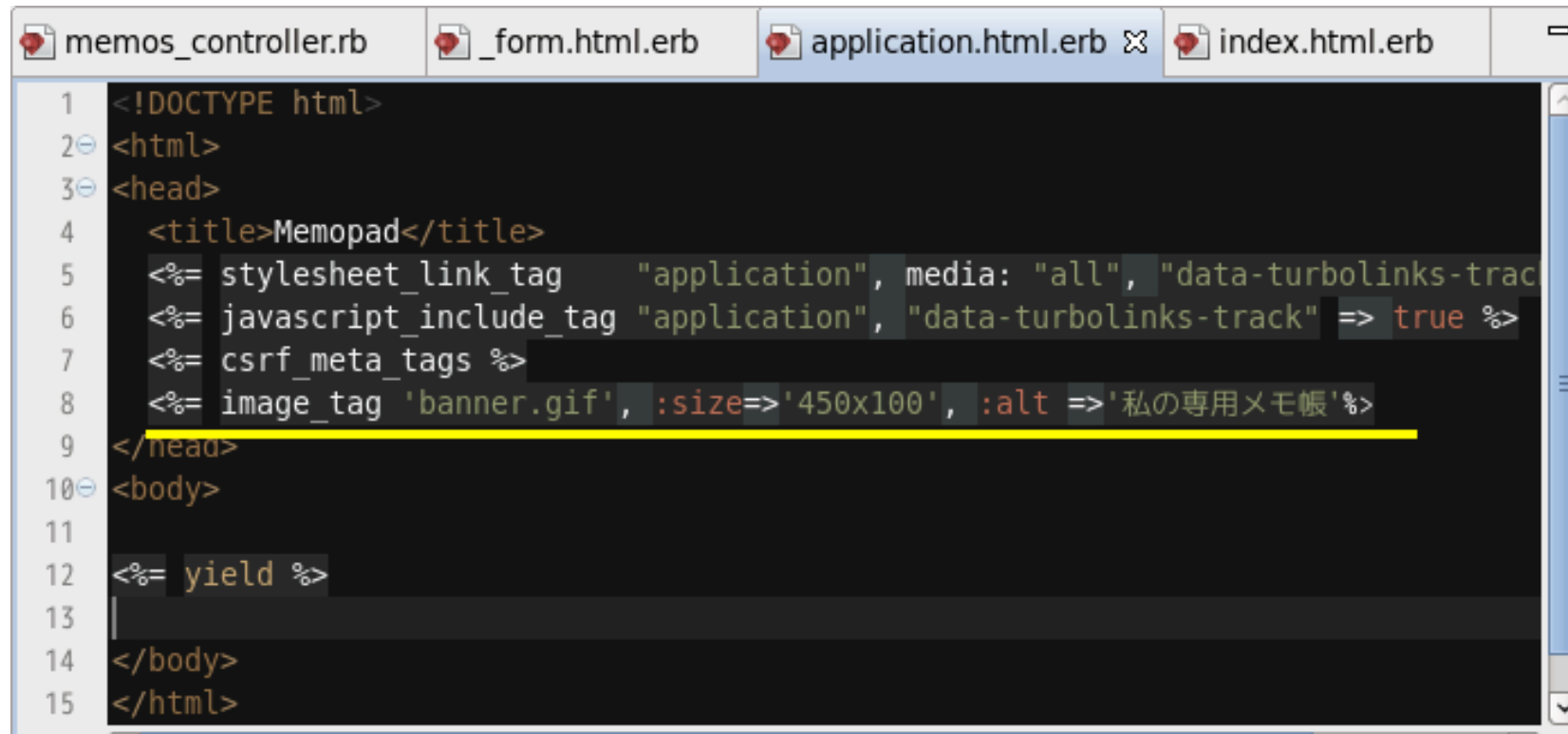
Windows側の D:/VMware/Shareと、Linux側の /mnt/hgfs/sharedとの間の共有設定を確認してください。

VMwareの「設定」画面の「共有」で設定できます。

Windows上でファイルをShareに書き込み、Linux側で読み出すことができます。

例題演習の画面

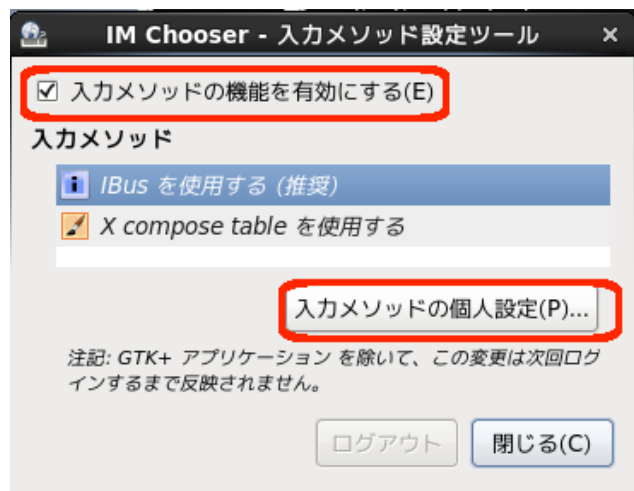
私の編集例です。



```
memos_controller.rb  _form.html.erb  application.html.erb  index.html.erb
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Memopad</title>
5   <%= stylesheet_link_tag "application", media: "all", "data-turbolinks-trac
6   <%= javascript_include_tag "application", "data-turbolinks-track" => true %>
7   <%= csrf_meta_tags %>
8   <%= image_tag 'banner.gif', :size=>'450x100', :alt =>'私の専用メモ帳'%>
9 </head>
10 <body>
11
12 <%= yield %>
13
14 </body>
15 </html>
```


日本語入力

「システム」→「設定」から
「入力メソッド」を選択します。

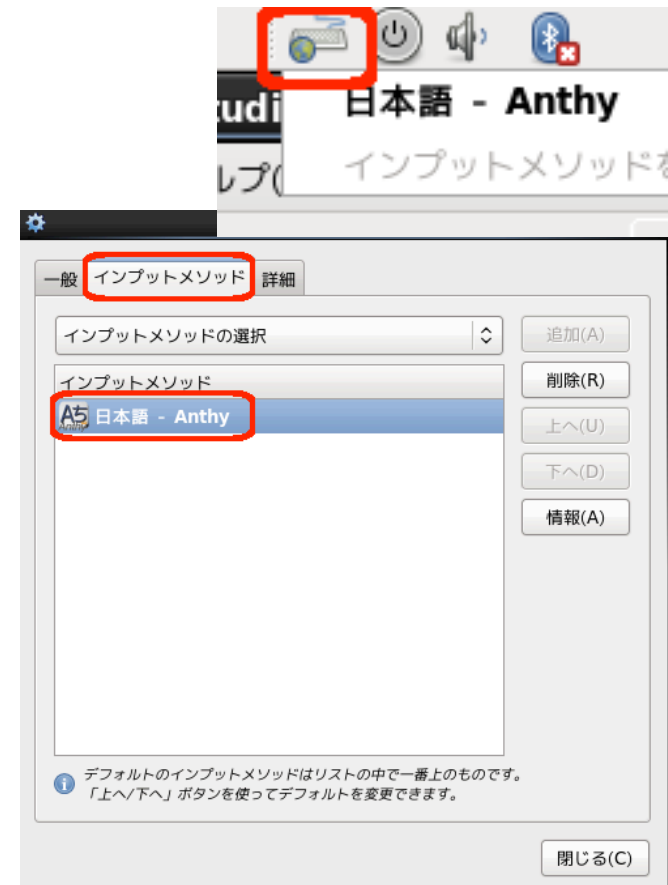


今回の環境での日本語入力

入力メソッドの個人設定を行うと、「日本語 - Anthy」というのがありますので、この設定や、アイコンを確認して下さい。

画面右上に、入力メソッドのアイコンが表示されています。

日本語入力の方法は
Anthyを調べて下さい。



今回の実行画面

上記の修正で、以下のような画面になりました。



画像表示のタグの例

```
<!DOCTYPE html>
<html>
<head>
  <title>Memopad</title>
  <%= stylesheet_link_tag "application", media: "all", "data-turbolinks-track" => true
    %>
  <%= javascript_include_tag "application", "data-turbolinks-track" => true %>
  <%= csrf_meta_tags %>
  <%= image_tag 'banner.gif', :size=>'450x100', :alt =>'私の専用メモ帳'%>
</head>
<body>

<%= yield %>

</body>
</html>
```

HTMLによる画像の表示

HTMLによる画像表示のタグは

```

```

である。この記述でも画像が貼付けられる。

この場合のファイルは、publicの下に置く…

はずだったが、railsのバージョンが変わって、
場所が変わった!?

リソース管理の都合で、なるべくrailsの機能を使って
下さい。

ソースコードを読んでみよう

プログラマにとって大切なことは、
長文のプログラムでも丁寧に読みこなすこと。

WEBページを
右クリックし、
「ソースを表示」を
選んで、
ソースコードを
読んでみよう。



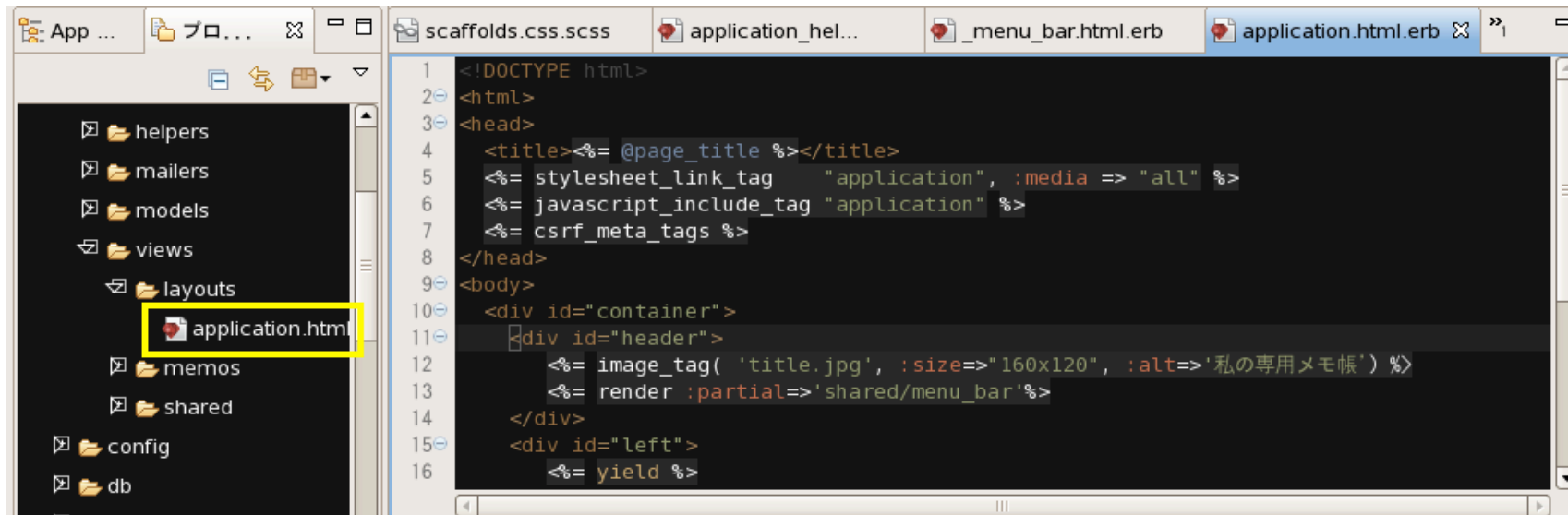
次のテーマ

CSS(Cascading Style sheet)を利用する。
画面全体を分割して、それぞれのスタイルを決める。
統一性のあるデザインとする。

【注意点】

様々なファイルを新規に追加していくので、それらの修正・追加作業が完結しないと、テストランは動きません。

Application.html.erb



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title><%= @page_title %></title>
5   <%= stylesheet_link_tag "application", :media => "all" %>
6   <%= javascript_include_tag "application" %>
7   <%= csrf_meta_tags %>
8 </head>
9 <body>
10  <div id="container">
11    <div id="header">
12      <%= image_tag( 'title.jpg', :size=>"160x120", :alt=>'私の専用メモ帳') %>
13      <%= render :partial=>'shared/menu_bar' %>
14    </div>
15    <div id="left">
16      <%= yield %>
```


Id として、属性名を与える

DOM (Document Object Model)で、画面上の要素一つ一つが「オブジェクト」として識別されています。

Javascriptなどでは、DOMのIDを経由して、画面上の要素一つ一つを動的に書き換えることができます。

ブロックレベルで指定しているidを、呼び出している側(.html.erb)と、定義している側(.css)とで対応付けて呼んでください。

同じ内容のテキストページ

```
<!DOCTYPE html>
<html>
<head>
  <title>Memopad</title>
  <%= stylesheet_link_tag "application", media: "all", "data-turbolinks-track" => true %>
  <%= javascript_include_tag "application", "data-turbolinks-track" => true %>
  <%= csrf_meta_tags %>
</head>
<body>
  <div id="container">
    <div id="header">
      <%= image_tag 'banner.gif', :size=>'450x100', :alt =>'私の専用メモ帳'%>
      <%= render :partial => 'shared/menu_bar' %>
    </div>
    <div id='left'>
      <%= yield %>
    </div>
    <div id='right'>
      <%= render :partial => 'shared/right_bar' %>
    </div>
    <div id='footer'>
      <%= render :partial => 'shared/footer' %>
    </div>
  </div>
</body>
</html>
```

Containerを定義し、Header/left/right/footerの4つの領域を作った。

Sharedのフォルダを作り、menu_barなどを記述する。これらの修正が終わらないと、テストランはできません。

1ブロックずつ徐々に追加し、テストランしながら進めてください。

Container (コンテナ)

内容全体を含むデザインを指定する。

内部に、

header

left

right

footer

の四つの構造を持たせる。

Leftに `<%= yield %>`

で、元々あった構造を持たせる。

開発時の注意

menu_bar

right_bar

footer

の三つの別ファイルを、sharedの下に配置する。

これらを書いてから、全体の枠指定を修正しないとファイルが見つからない、というエラーになります。

また、`<div>` と `</div>` の入れ子構造に注意する。

タイトルバナーの移動

bannarファイルを、headブロックからbodyブロック内部の、head 部分に移動しました。

画面の表示するものは、bodyブロックにまとめるようにしました。

HTMLでのコメントアウト

`<%= render :partial => 'shared/menu_bar' %>`
で、まだmenu_barを記述していないため、この1行
をコメントアウトします。

HTMLのコメントは、

`<!-- -->`

です。このタグをコメントにするため、

`<!-- %= render :partial => 'shared/menu_bar' % -->`

にしました。

CSS(スタイルシート)の指定

```
<div id="container">
```

では、containerという名前のスタイルを指定しました。

スタイルは、スタイルシート(Cascading Style Sheet)のファイルに記述します。スタイルシートは

`app¥assets¥stylesheets`

フォルダにあります。

今回は、`scaffold.css`を修正します。

対応しているscaffold.cssを修正

□ 以下の内容を、そのまま書いて見る

```
div#container{  
  background-color: white;  
  margin: 0 auto;  
  padding: 5px 0 0;  
  width: 800px;  
}
```

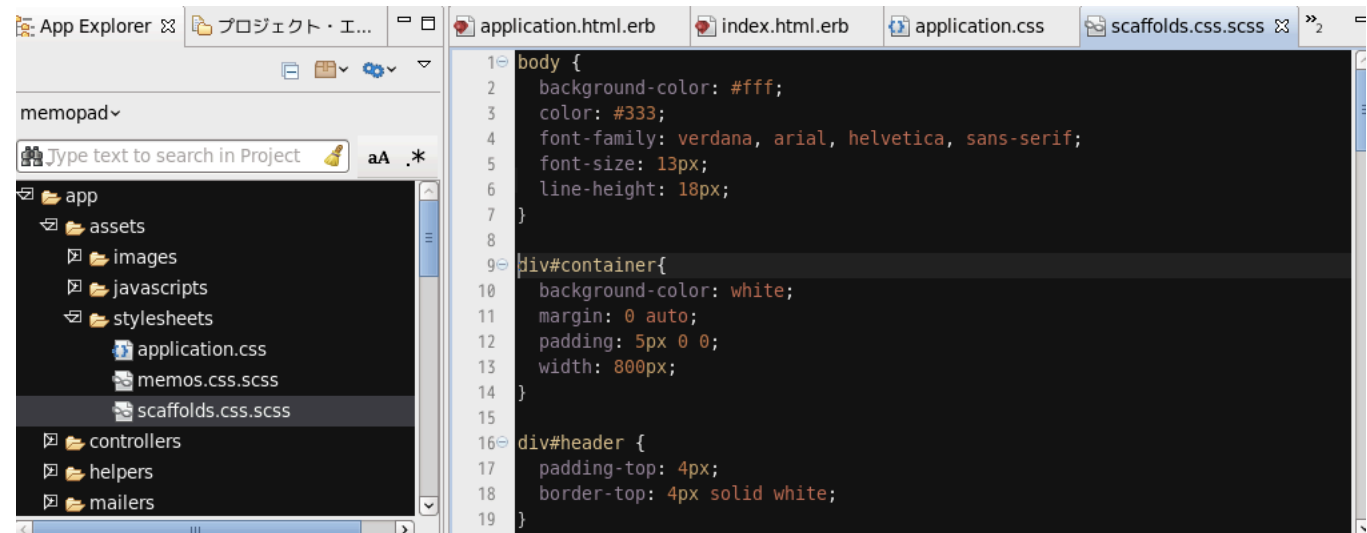
```
div#header {  
  padding-top: 4px;  
  border-top: 4px solid white;  
}
```

```
div#left {  
  float: left;  
  background-color: white;  
  padding: 6px 6px 6px 0;  
}
```

```
div#right {  
  float: right;  
  width: 228px;  
  background-color: #e8ffff;  
}
```

```
div#footer {  
  background-color: white;  
  border-top: 2px lightgreen solid ;  
  text-align: center;  
  clear: both;  
  width: 100%;  
}
```

```
div#menu_bar{  
  color: black;  
}
```



scaffold.css.scss全体

```
body {
  background-color: #fff;
  color: #333;
  font-family: verdana, arial, helvetica, sans-serif;
  font-size: 13px;
  line-height: 18px;
}

div#container{
  background-color: white;
  margin: 0 auto;
  padding: 5px 0 0;
  width: 800px;
}

div#header {
  padding-top: 4px;
  border-top: 4px solid white;
}

div#left {
  float: left;
  background-color: white;
  padding: 6px 6px 6px 0;
}

div#right {
  float: right;
  width: 228px;
  background-color: #e8ffff;
}

div#footer {
  background-color: white;
  border-top: 2px lightgreen solid;
  text-align: center;
  clear: both;
  width: 100%;
}

div#menu_bar{
  color: black;
}

p, ol, ul, td {
  font-family: verdana, arial, helvetica, sans-serif;
  font-size: 13px;
  line-height: 18px;
}

pre {
  background-color: #eee;
  padding: 10px;
  font-size: 11px;
}

a {
  color: #000;
  &:visited {
    color: #666;
  }
  &:hover {
    color: #fff;
    background-color: #000;
  }
}

div {
  &.field, &.actions {
    margin-bottom: 10px;
  }
}

#notice {
  color: green;
}

.field_with_errors {
  padding: 2px;
  background-color: red;
  display: table;
}

#error_explanation {
  width: 450px;
  border: 2px solid red;
  padding: 7px;
  padding-bottom: 0;
  margin-bottom: 20px;
  background-color: #f0f0f0;
  h2 {
    text-align: left;
    font-weight: bold;
    padding: 5px 5px 5px 15px;
    font-size: 12px;
    margin: -7px;
    margin-bottom: 0px;
    background-color: #c00;
    color: #fff;
  }
  ul li {
    font-size: 12px;
    list-style: square;
  }
}
```

入力時の注意とお願い

この教材は、一人でも多くの人々がテストランに成功するように、テキストでプログラムをそのまま掲載しています。これを、コピーしても動作はします。

ただ、「意味」を考えないと理解にはつながらないので、各自の責任で意味を考えてください。

また、PDFなどで[00]などの制御文字が入り、それが見えないままエラーの原因となるケースが散見されます。「空白」や「改行」を削除して、「空白」や「改行」を入力し直すと、このエラーは回避できます。

memos/index.html.erbと layouts/application.html.erbの関係

- Application.html.erbの<%= yeild %>
 - に、他のindex.html.erbの中身が展開される。
- memosは、指定されたURL
 - ここでは、http://127.0.0.1:3000/
 - 例えば、http://watacinomemo.org/などのように
 - 「看板」のURLが入り、applicationはその全体に共通するレイアウトとなる。
- デザインに一貫性を持ちたい部分をapplicationに。
 - サブ画面として、個性を持たせ、区別したいそのサブ画面のアイテムはmemos/index.html.erbで記述を行う。

メニューバーを入れてみる。

□ 最初に、修正の全体像を把握する

(1) application.html.erbに1行追加

この記述で、menu_barが展開される。

```
<%= render :partial => 'shared/menu_bar' %>
```

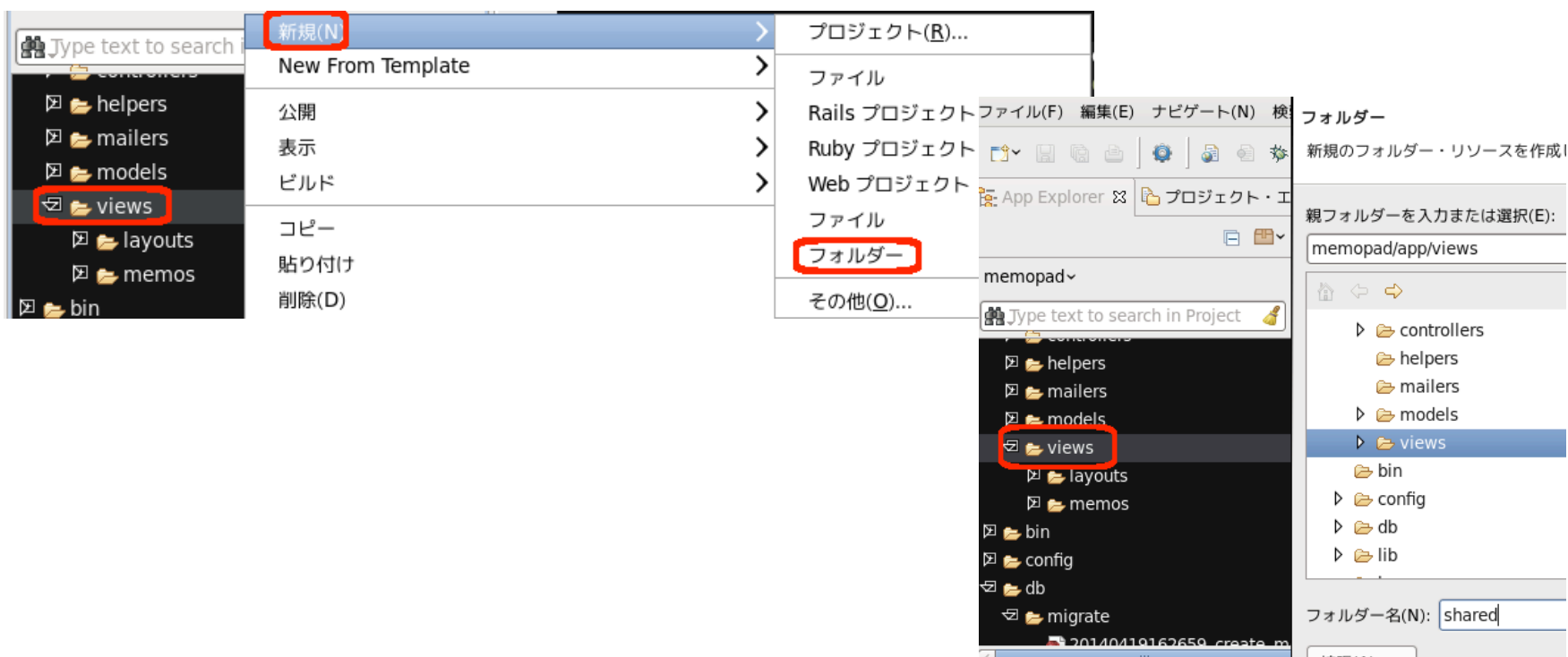
(2) viewsにsharedフォルダを作り、
_menu_bar.html.erbを書く。

(3) _menu_bar.html.erbで呼ばれている
menu_link_to(item)メソッドを、
application_helper.rb内に書く。

サブフォルダの作成

Viewsの上で右クリックして、「新規」→「フォルダ」と入力する。

sharedというフォルダ名を入力する。



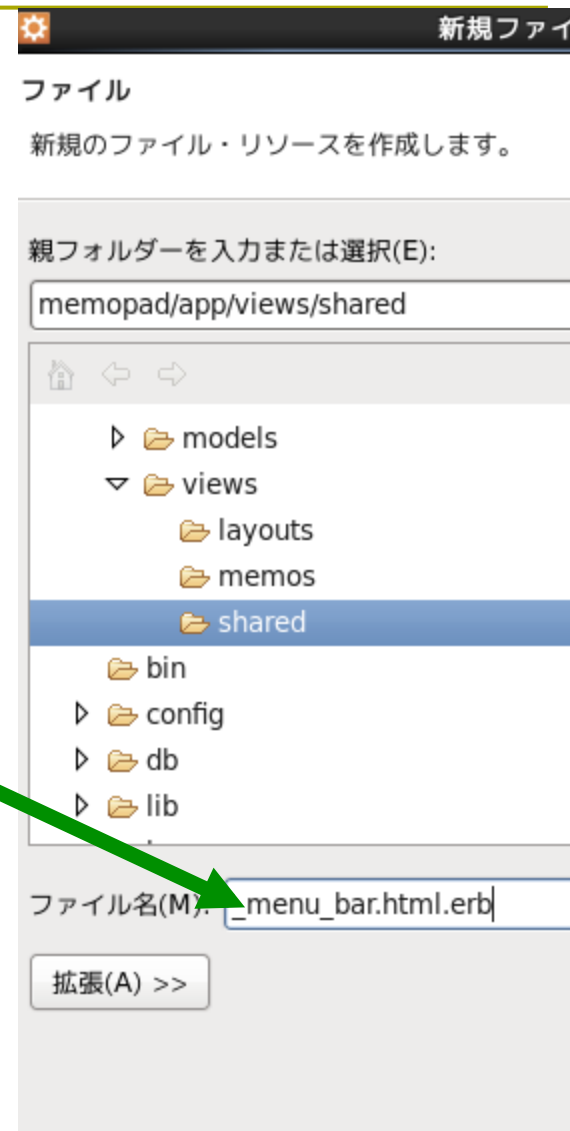
menu_barの呼び出し

- views/layouts/application.html.erbに1行追加
 - `<%= image_tag...`の下
- ```
<%= render :partial => 'shared/menu_bar' %>
```
- renderは、htmlへの展開
  - :partialで部分ファイル指定

```
8 </head>
9 <body>
10 <div id="container">
11 <div id="header">
12 <%= image_tag 'banner.gif', :size=>'450x100', :alt =>'私の専用メモ帳'%>
13 <%= render :partial => 'shared/menu_bar' %>
14 </div>
15 <div id='left'>
16 <%= yield %>
17 </div>
18 <div id='right'>
19 <!-- %= render :partial => 'shared/right_bar' % -->
--
```

# \_menu\_bar.html.erbを作る

- sharedフォルダで右クリックし「新規」⇒「ファイル」へ
- ファイル名を入力する。  
\_menu\_bar.html.erb



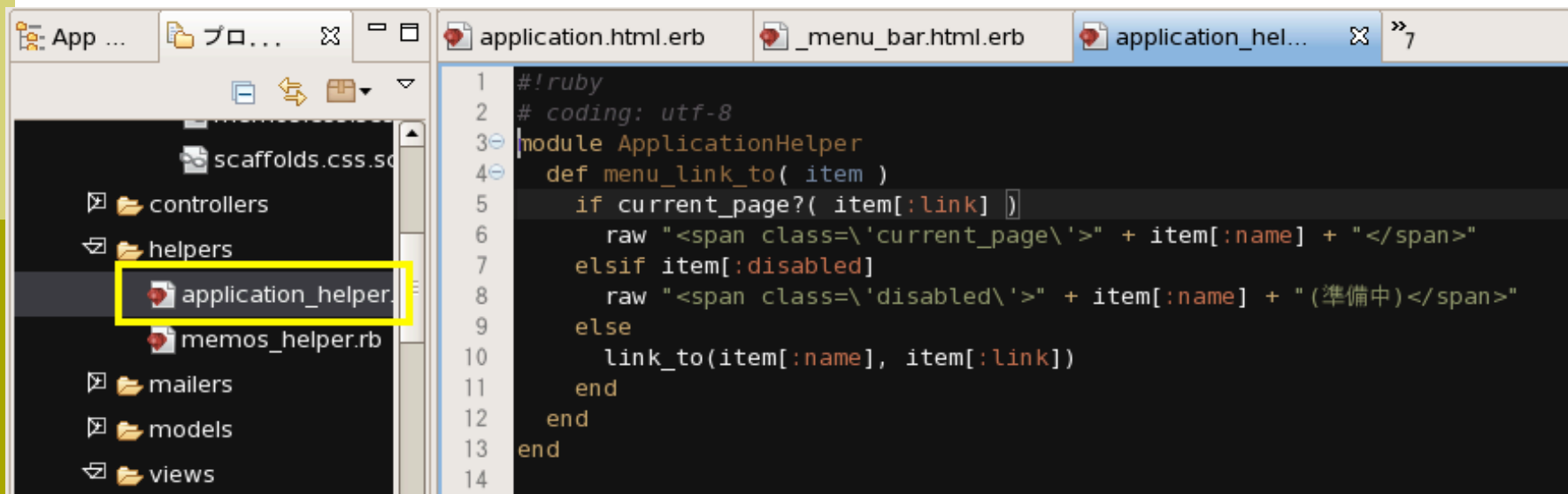






# 最後に、menu\_link\_toの記述

- application\_helper.rbに、
  - menu\_link\_to メソッドを記述する。



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure with folders for controllers, helpers, mailers, models, and views. The application\_helper.rb file is highlighted in the helpers folder. The code editor shows the implementation of the menu\_link\_to method in application\_helper.rb.

```
1 #!ruby
2 # coding: utf-8
3 module ApplicationHelper
4 def menu_link_to(item)
5 if current_page?(item[:link])
6 raw "" + item[:name] + ""
7 elsif item[:disabled]
8 raw "" + item[:name] + "(準備中)"
9 else
10 link_to(item[:name], item[:link])
11 end
12 end
13 end
14
```

# 同じページのテキスト記述

```
#!/ruby
coding: utf-8
module ApplicationHelper
 def menu_link_to(item)
 if current_page?(item[:link])
 raw "" + item[:name] + ""
 elsif item[:disabled]
 raw "" + item[:name] + "(準備中)"
 else
 link_to(item[:name], item[:link])
 end
 end
end
end
```

PDFでは、\'current\_page  
\'と\'disabled\'の両端  
の\'(引用符)が全角になってい  
ます。注意してください。

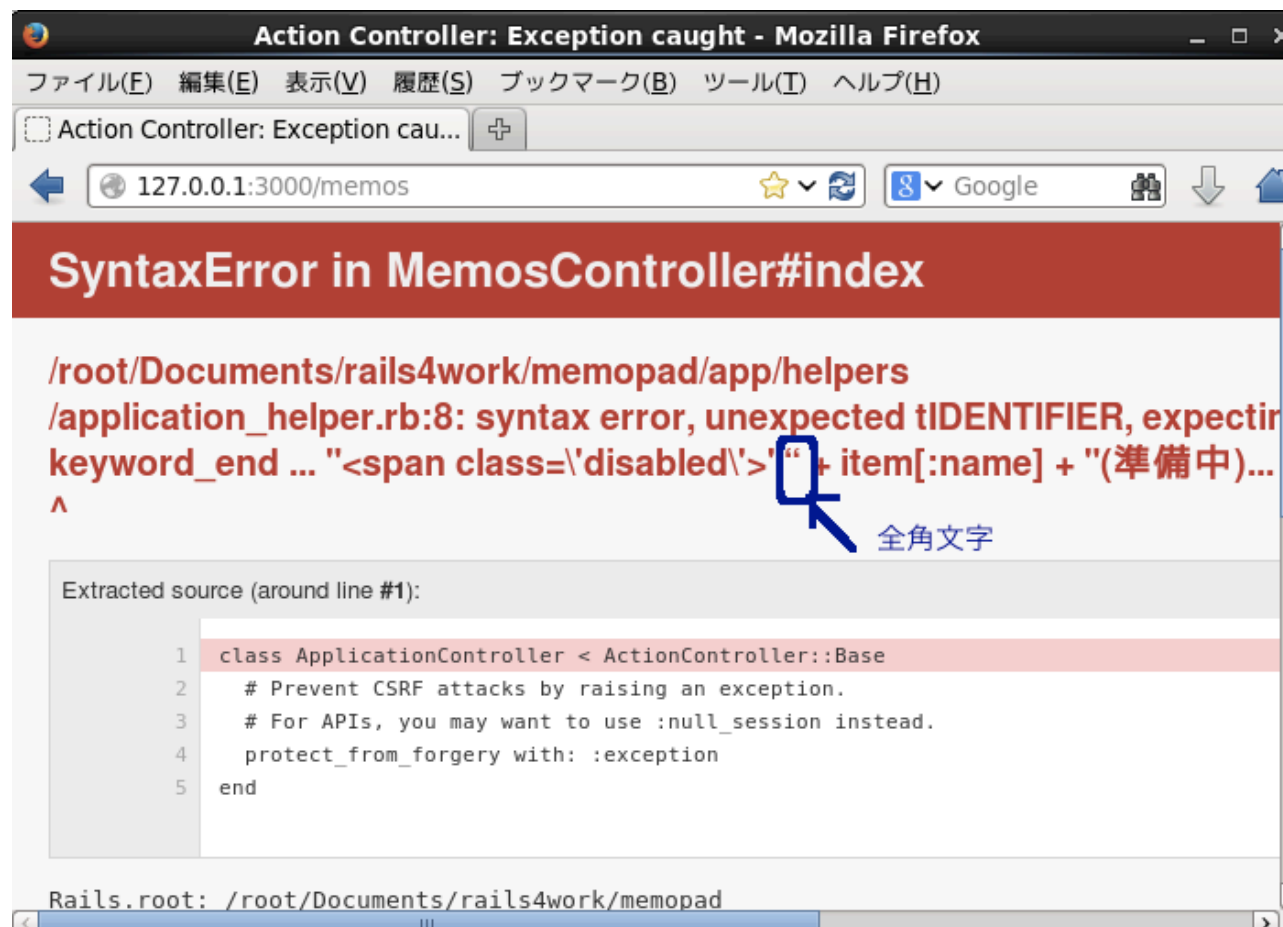
# ここまでの画面

- タイトル画像とメニューバーを、上下2分割のレイアウトで入れてみた。



# トラブルシューティング

こんなのが出てしまった・・・



# 全角文字によるエラー

---

コンソール画面を参照し、エラーのある行を探してください。このPDFの' (引用符) は全角文字として表示されています。

全角文字によるエラーは、場所により出方が異なります。

注意してメッセージを読んで下しさい。

# ファイルを準備しないでテストラン

Right\_barを準備しないうちに、引用した場合

```
<div id="right">
 <%= render :partial=> 'shared/right_bar' %>
</div>
```

この部分を削るか、  
コメントアウトすれば  
解決します。

```
Action Controller: Exception caught - Mozilla Firefox
ファイル(E) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
Action Controller: Exception cau...
127.0.0.1:3000/memos
ActionView::MissingTemplate in Memos#index
Showing /root/Documents/rails4work/memopad/app/views/layouts/application.html.erb where line #19 raised:
Missing partial shared/right_bar with {:locale=>[:en], :formats=>[:html], :handlers=>[:erb,
:builder, :raw, :ruby, :jbuilder, :coffee]}. Searched in:
* "/root/Documents/rails4work/memopad/app/views"
Extracted source (around line #19):
16 <%= yield %>
17 </div>
18 <div id='right'>
19 <%= render :partial => 'shared/right_bar' %>
20 </div>
21 <div id='footer'>
22 <!-- %= render :partial => 'shared/footer' % -->
```

# CSSとは！

---

- 今更ながら、CSSとは・・・
- Cascading Style Sheet
- WEBページを簡便に、体系的に見やすく修飾するための簡易言語
  
- ↓ このページがわかりやすい。
- <http://www.htmq.com/csskihon/001.shtml>

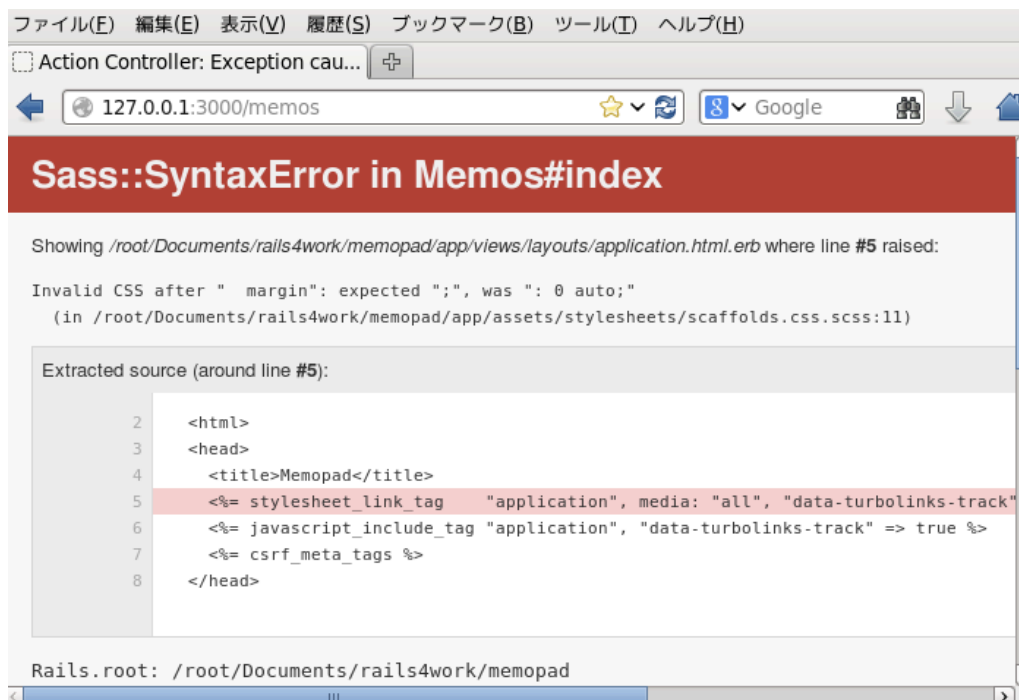


# CSSで ; を忘れると・・・

こんなエラーが出ました。

(まったく同じエラーになることは保証しませんが・・・)

cssではなくhtml側でエラーになっている点に注意して下さい。



The screenshot shows a web browser window with a red error banner at the top that reads "Sass::SyntaxError in Memos#index". Below the banner, the browser displays the following error message:

```
Showing /root/Documents/rails4work/memopad/app/views/layouts/application.html.erb where line #5 raised:
Invalid CSS after " margin": expected ";", was ": 0 auto;"
(in /root/Documents/rails4work/memopad/app/assets/stylesheets/scaffolds.css.scss:11)
```

Below the error message, the browser shows the "Extracted source (around line #5):" section with the following HTML code:

```
2 <html>
3 <head>
4 <title>Memopad</title>
5 <%= stylesheet_link_tag "application", media: "all", "data-turbolinks-track"
6 <%= javascript_include_tag "application", "data-turbolinks-track" => true %>
7 <%= csrf_meta_tags %>
8 </head>
```

The error message and the HTML code are displayed in a light gray background. The browser's address bar shows "127.0.0.1:3000/memos" and the title bar shows "Action Controller: Exception cau...". The browser's status bar at the bottom shows "Rails.root: /root/Documents/rails4work/memopad".

# CSSを変えると...

The image shows a code editor on the left and a browser window on the right. The code editor displays the following CSS code:

```
1 body {
2 background-color: #fff;
3 color: #333;
4 font-family: verdana, arial, helvetica, sans-serif;
5 font-size: 13px;
6 line-height: 18px;
7 }
8
9 div#container{
10 background-color: blue;
11 margin: 0 auto;
12 padding: 5px 0 0;
13 width: 800px;
14 }
15
16 div#header {
17 padding-top: 4px;
18 border-top: 4px solid black;
19 }
```

The browser window, titled "Memopad - Mozilla Firefox", shows the rendered page at `127.0.0.1:3000/memos`. The page features a blue background for the main content area, a header with a black border, and a navigation bar with links: "一覧表示", "新規追加", "ruby 公式サイト", and "自分の課題ページ(準備中)". The main content area displays "メモの一覧" and "メモ見出し 作成者".

# Rightとfooter

---

`_right_bar.html.erb`

「広告募集中」とだけ書きました。

`_footer.html.erb`

Copyright (C) by Ikuo.Kobayashi  
と書いてみました。

# 今日の最終画面

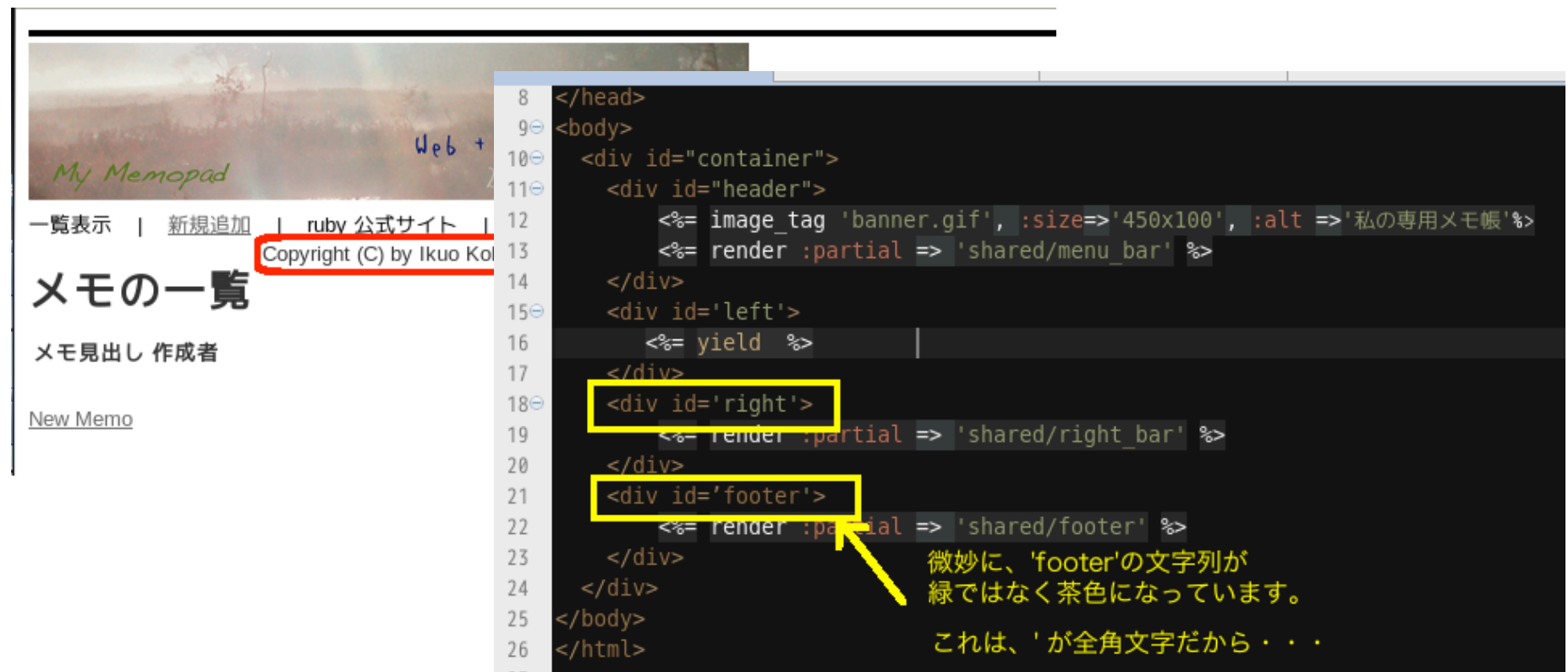
こんな感じです。



# フッタがおかしい！

思い通りの表示にならない・・・

エラーにならないミスは、探すのが大変です。



```
8 </head>
9 <body>
10 <div id="container">
11 <div id="header">
12 <%= image_tag 'banner.gif', :size=>'450x100', :alt =>'私の専用メモ帳' %>
13 <%= render :partial => 'shared/menu_bar' %>
14 </div>
15 <div id='left'>
16 <%= yield %>
17 </div>
18 <div id='right'>
19 <%= render :partial => 'shared/right_bar' %>
20 </div>
21 <div id='footer'>
22 <%= render :partial => 'shared/footer' %>
23 </div>
24 </div>
25 </body>
26 </html>
```

Copyright (C) by Ikuo Koiwai

メモの一覧

メモ見出し 作成者

New Memo

微妙に、'footer'の文字列が緑ではなく茶色になっています。

これは、'が全角文字だから・・・

# 今日のレポート課題

---

- 今日は、ありません。
- どんな「画面」にデザインしたいか、参考になりそう  
WEBサイトを検索したり、デザインの雑誌を読んだりして、イメージを固めてください。

# きょうの授業を休んだ人

---

- 自分なりの画像ファイルなどを用意して、CSSを組み込んで下さい。
- 欠席課題のレポートには、application.html.erbのソースコードと、それ以外に実際に修正したファイル、CSSのプログラム、index.html.erbのプログラムなどを添付(テキスト貼り付けOK)し、画面を貼り付けて報告して下さい。
- この「欠席課題」の提出があれば、「出席扱い」に切り替えます。

# 参考ページ

---

- 超初心者のためのホームページ作成講座
- 超初心者のためのスタイルシート講座
- <http://park16.wakwak.com/~html-css/index.html>
  
- スタイルシート・レファレンス
- <http://www.htmq.com/style/index.shtml#tex>