

EPI: Interfejs Graficzny 2011/2012

Laboratorium nr 4

Model-Widok-Kontroler

Aleksander Pohl

7 grudnia 2011

Plan prezentacji

Książka gości

Zadanie

Zaimplementować funkcjonalność książki gości, bez użycia rusztowania.

Model Entry

1. ▷ rails g model entry name:string comment:text
2. ▷ rake db:migrate
3. Otwieramy plik **app/models/entry.rb**:

```
class Entry < ActiveRecord::Base
  validates_presence_of :name, :comment
end
```

Kontroler i trasowanie

1. ▷ rails g controller entries index new create
2. modyfikujemy plik **config/routes.rb**

```
Library::Application.routes.draw do
  # usuwamy:
  # get 'entries/index'
  # get 'entries/new'
  # get 'entries/create'
  resources :entries, :only => [:index, :new, :create]

  root :to => "entries#index"
end
```

3. usuwamy plik **public/index.html**
4. definiujemy 3 metody:
 - ▶ index – która wyświetla wszystkich gości
 - ▶ new – która wyświetla formularz do tworzenia wpisów
 - ▶ create – która zachowuje wpis w bazie danych

Kontroler EntriesController

app/controllers/entries_controller.rb

```
# encoding: utf-8
class EntriesController < ApplicationController
  def index
    @entries = Entry.order("created_at DESC")
  end

  def new
    @entry = Entry.new
  end

  def create
    @entry = Entry.new(params[:entry])
    if @entry.save
      flash[:notice] = "Dziękujemy za wpis w książce gości!"
      redirect_to :action => "index"
    else
      render :action => "new"
    end
  end
end
```



Widok – index

Otwieramy plik: `app/views/entries/index.html.erb`

```
<h1>Książka gości biblioteki</h1>
<% if flash[:notice] %>
  <div class="notice"><%= flash[:notice] %></div>
<% end %>
<p>Ostatnie wpisy:</p>
<% @entries.each do |entry| %>
  <div>
    <div><%= entry.name %> napisał(a) <%= entry.created_at %>:</div>
    <div><%= entry.comment %></div>
  </div>
<% end %>

<br />
<%= link_to "Nowy wpis", new_entry_path %>
```

Widok – new

Otwieramy plik: `app/views/entries/new.html.erb`

```
<h1>Nowy wpis w księdze gości</h1>
<%= form_for @entry do |f| %>
  <% if @entry.errors.any? %>
    <ul>
      <% @entry.errors.full_messages.each do |msg| %>
        <li><%= msg %></li>
      <% end %>
    </ul>
  <% end %>
<p>
  <%= f.label :name, "Twoje imię:" %><br/>
  <%= f.text_field :name %>
</p>
<p>
  <%= f.label :comment, "Twój komentarz:" %> <br/>
  <%= f.text_area :comment %>
</p>
<%= f.submit "Dodaj" %>
<% end %>
```



Zadanie

- ▶ zaimplementuj w interfejsie użytkownika aplikacji **Portfel** zamianę waluty
 - ▶ ustaw odpowiednie trasowanie
 - ▶ dodaje dwie akcje do kontrolera Money: `exchange_form` oraz `exchange`
 - ▶ dodaj formularz zamiany jednej waluty na inną
 - ▶ dodaj link w widoku portfela pozwalający na zamianę waluty
 - ▶ ustal współczynnik zamiany
 - ▶ oblicz ilość waluty docelowej
 - ▶ zmniejsz ilość waluty źródłowej
 - ▶ w zależności od tego czy waluta docelowa istnieje: utwórz lub zmodyfikuj ilość waluty docelowej
 - ▶ w modelu Money zaimplementuj metodę zamiany waluty
 - ▶ zaimplementuj akcje `exchange_form` oraz `exchange` posługując się nową metodą modelu

Zadania – cd

- ▶ utwórz layout z linkami do portfela i tablicy kursów oraz lepszym formatowaniem
- ▶ zmodyfikuj widok portfela
 - ▶ usuń link *show* oraz *destroy*
 - ▶ zastąp link *edit* linkiem obejmującym wartość i walutę kursu
 - ▶ popraw wygląd poszczególnych pozycji edytując widok i akrusz styli
- ▶ zmodyfikuj widok tabeli kursów
 - ▶ usuń link *show* oraz *destroy*
 - ▶ zastąp link *edit* linkiem obejmującym walutę źródłową i docelową
 - ▶ popraw wygląd poszczególnych pozycji
 - ▶ spraw by wyświetlane były zawsze 3 miejsca po przecinku
- ▶ zmodyfikuj url aplikacji (w konfiguracji trasowania), tak aby domyślnie wyświetlany był portfel