02NPYPD - LINGUAGGI E AMBIENTI MULTIMEDIALI A

PAGINE, HELPERS AND ROUTES

Creazione del progetto "PoliRun" e delle prime pagine HTML

La seconda esercitazione ha lo scopo di creare il progetto "PoliRun" – il social network che svilupperemo durante le esercitazioni di laboratorio – con le prime pagine HTML, il primo helper e le prime route.

ESERCIZIO 1 - PAGINE STATICHE

Dopo aver creato un nuovo progetto Rails con RubyMine, generare un controller per gestire le pagine "statiche" del nostro social network. Tale controller, che chiameremo *Pages*, avrà tre pagine associate: *Home, About e Contact*. Inserire nella home dei link alle altre pagine, utilizzando il costrutto `link_to` di Rails.

ESERCIZIO 2 - UN PO' DI DINAMICITÀ

L'obiettivo di questo esercizio è quello di impostare il titolo delle pagine HTML5 create nell'esercizio precedente a "PoliRun | *Nome_della_pagina*", senza ripetere la struttura HTML necessaria per farlo in ogni singola pagina.

Per far ciò, utilizzeremo la funzione `provide` all'inizio di ogni vista, associando al simbolo `:title` il nome della pagina corrente. Utilizzeremo poi questo simbolo direttamente nel tag title HTML5 presente in `application.html.erb` (file che definisce la struttura di una generica pagina all'interno dell'intera applicazione).

ESERCIZIO 3 - IL PRIMO HELPER

Arrivati a questo punto, ogni pagina del progetto avrà un titolo diviso in due parti: una statica ("PoliRun | ") e una dinamica (il nome della pagina corrente). Cosa accadrebbe se ci dimenticassimo di inserire il simbolo `:title` in una nuova vista?

Tenendo presente che un helper si scrive esattamente come un metodo (`def... end`), costruire un helper che:

- abbia `full_title` come nome e prenda come parametro il titolo della pagina (`page_title`);
- mostri il titolo delle pagine come avviene adesso, ma solo se `page_title` esiste;
- mostri solo la parola "PoliRun" se la variable `page_title` non esiste.

Gli helper che riguardano tutta l'applicazione web vanno inseriti nel file `/app/helpers/application_helper.rb`. Per usare l'helper, nel nostro caso, ci basterà poi sostituire il contenuto di `<title>...</title>` con

```
<title><%= full_title(yield(:title)) %></title>
```

ESERCIZIO 4 - ROUTES

Assegnare delle (named) routes alle pagine esistenti (home, contact, about), agendo sul file `routes.rb` (dentro la cartella `config`). Vogliamo far sì che la home sia raggiungibile direttamente digitando nel browser (ad esempio) `http://localhost:3000/`, e che le altre pagine siano figlie dirette di tale URL (es. `http://localhost:3000/about`).

In altre parole, vorremmo ottenere quanto riportato in tabella:

Page	URI	Named route
Home	/	root_path
About	/about	about_path
Faq	/faq	faq_path
Contact	/contact	contact_path
Sign up	/signup	signup_path
Sign in	/signin	signin_path

Una *named route* si definisce sostituendo i getter delle singole pagine con

```
match '/nome_della_pagina', to: 'nome_del_controller#nome_della_pagina'
```

La sintassi per la pagina di root (la home, nel nostro caso) è, invece, `root to: 'nome_controller#nome_pagina'`. Affinché la route funzioni, in questo caso, occorre cancellare o rinominare il file `index.html` presente dentro la cartella `public`.

Aggiornare, a questo punto, i vari link presenti nella homepage (es. `<%= link_to "About", '#' %>`) indicando le named routes appena dichiarate (es. `<%= link_to "About", about_path %>`).

Creare, infine, la pagina `faq`, adeguarla alle altre pagine e impostargli una named route.

APPROFONDIMENTO TEORICO

Come riportato nel set di slide “Rails101”, ogni volta che nel browser si digita un indirizzo che, subito dopo la radice, contiene il nome di una pagina, come `/about`, viene controllato il file `routes.rb` alla ricerca di una corrispondenza. Se tale corrispondenza esiste come una route del tipo `match '/about', to: 'pages#about'` allora viene richiamata l’azione corrispondente (`about`, in questo caso) definita all’interno del suo controllore (`Pages`).

Inoltre, il codice `match '/about'` crea automaticamente le seguenti variabili che possono essere utilizzate nei controllori e nelle viste per richiamare le giuste pagine:

```
about_path => '/about'
about_url  => 'http://localhost:3000/about'
```