lenart / **admin_lte_rails**

Rails gem for AdminLTE admin theme

64 commits | **1** branch | **0** packages | **2** releases | **2** contributors

Branch: **master** | **New pull request** | **Create new file** | **Upload files** | **Find file** | **Clone or download**

lenart Merge branch 'master' of github.com:lenart/admin_lte_rails ··· Latest commit `1cf1fce` on Jun 5

app	Added width to reordable column.	3 years ago
bin	Initial import	5 years ago
lib	Replace render nothing: true with head	7 months ago
.gitignore	Initial import	5 years ago
.travis.yml	Initial import	5 years ago
Gemfile	Updates	5 years ago
README.md	Add code for reordering. Update readme	4 years ago
Rakefile	Initial import	5 years ago
admin_lte_rails.gemspec	Update AdminLTE. Change quotes	4 years ago

README.md

AdminLTE for Rails

This is an attempt to create a reusable gem for AdminLTE theme.

Don't use this gem since it's changing a lot at the moment.

You might be better off using adminlte-rails or adminlte2-rails gem - at least for now.

This gem uses `rails-assets-admin-lte` gem under the hood. It started out as a container for AdminLTE assets but this is now handled for us by bower package at rails-assets.org.

This gem will probably slowly transform to only include helpers and custom modifications of AdminLTE.

Installation

Add this line to your application's Gemfile:

```
gem "admin_lte_rails"
```

And then execute:

```
$ bundle install
```

Usage

The easiest way to get started is to generate layout views

```
rails generate admin_lte_rails:views
```

Then in your controller define `layout 'admin_lte'` to use the new theme.

For widgets and their options check out theme's homepage.

Choosing admin theme

By default there no admin theme is used. You should include one from `rails-assets-admin-lte` gem. In your assets/application.scss add

```
//= require "admin-lte/skins/skin-blue"
```

Controller helpers

To get some default REST actions include `AdminLteRails::RestControllerConcern` in your controller.

```
class PostsController < AdminController
  include AdminLteRails::RestControllerConcern
end
```

You will probably override default actions for more complex actions.

View helpers

You can use `nav_link_to(text, url, icon=nil)` for generating links for sidebar.

Sidebar example

```
= content_for :admin_sidebar do
  %ul.sidebar-menu
    %li.header Products
    = nav_link_to "Products", [:admin, :products], "fa-bicycle"
    = nav_link_to "Categories", [:admin, :categories], "fa-list"
    = nav_link_to "Brands", [:admin, :brands], "fa-heart"
    = nav_link_to "Tags", [:admin, :tags], "fa-tags"
    %li.header Content
    = nav_link_to "Pages", [:admin, :pages], "fa-files-o"
    = nav_link_to "Blog posts", [:admin, :posts], "fa-newspaper-o"
```

Topbar example

```
= content_for :admin_topbar do
  %ul.nav.navbar-nav
    %li.dropdown.notifications-menu
      %a.dropdown-toggle{href:"#", "data-toggle" => "dropdown"}
        %span{class: "flag-icon flag-icon-#{I18n.locale}"}
      %ul.dropdown-menu
        %li
          %ul.menu
            %li{class: "#{'active' if I18n.locale == :en}"}
              = link_to url_for(locale: :en) do
                %span.flag-icon.flag-icon-en
                English
            %li{class: "#{'active' if I18n.locale == :de}"}
              = link_to url_for(locale: :de) do
                %span.flag-icon.flag-icon-de
                Deutsch
    %li
      - if signed_in?
        = link_to sign_out_path, method: :delete do
          %i.fa.fa-power-off
          %span Sign out
```

Template regions

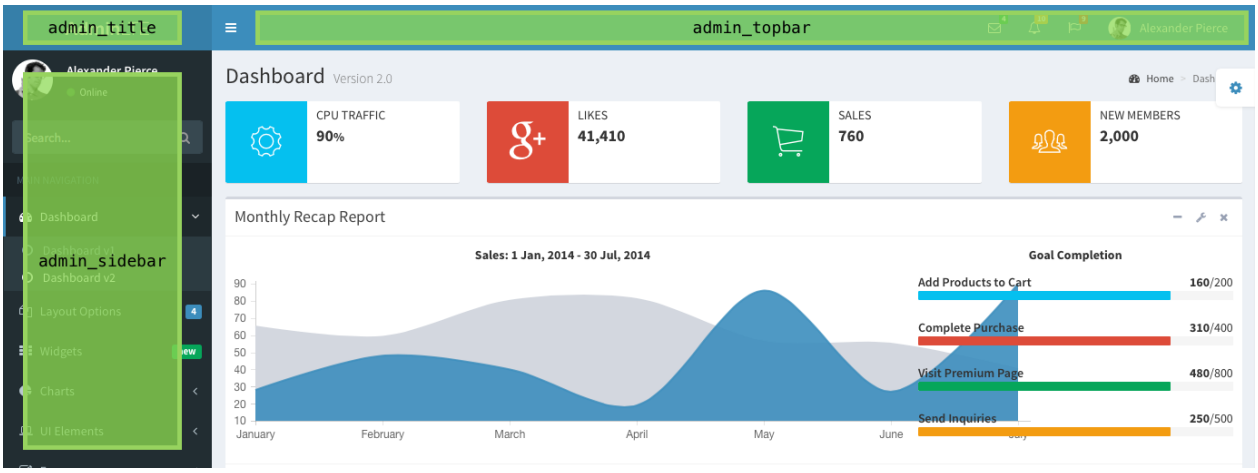
You can use `content_for` with the following regions:

- `content_for(:admin_title)`
- `content_for(:admin_topbar)`

- `content_for(:admin_sidebar)`

For adding extra stylesheets/javascripts use

- `content_for(:stylesheets)` - added to the head after theme stylesheets
- `content_for(:javascripts)` - added to the end of page's body after themes javascripts



Plugins

If you want to use plugins you have to manually load them.

Setup example for iCheck plugin:

```
# config/initializers/assets.rb
Rails.application.config.assets.precompile += %w(
  jquery-icheck.js jquery-icheck/skins/all.css
)
```

Add plugin to your Gemfile

```
  source 'https://rails-assets.org'
  gem 'rails-assets-jquery-icheck'
```

If you're using gems default layout you'll have to add assets by using `content_for :stylesheets` and `content_for :javascripts`.

```
# views/sessions/_form.html.haml
- content_for :stylesheets do
  = stylesheet_link_tag 'jquery-icheck/skins/all'

- content_for :javascripts do
  = javascript_include_tag 'jquery-icheck'
  :javascript
    $(function () {
      $('input').iCheck({
        checkboxClass: 'icheckbox_square-blue',
        radioClass: 'iradio_square-blue',
        increaseArea: '20%' // optional
      });
    });
```

Reordering items

Add Sortable gem to your Gemfile

```
source "https://rails-assets.org" do
  gem "rails-assets-Sortable"
end
```

Include javascripts in your admin.js

```
#= require Sortable/jquery.binding
#= require Sortable/Sortable
#= require admin_lte_rails/reordable
```

In your route file

```
concern :reordable do
  post "reorder", on: :collection
end

resources :posts, concerns: :reordable
```

In your stylesheets require

```
//= require admin_lte_rails/reordable
```

In your views make sure you have

```
tbody.ordable-list data-reorder-url=reorder_admin_categories_path
  - @categories.each do |category|
    tr data-record-id=category.id
      td.text-center.reorder-handle
        span.fa.fa-sort
      td
        = category.name
```

Toastr (growl-like notifications)

Include following in `application.js`:

```
//= require toastr
```

Include following in `application.css`:

```
*= require toastr
```

Contributing

1. Fork it (https://github.com/lenart/admin_lte_rails/fork)
2. Create your feature branch (`git checkout -b my-new-feature`)
3. Commit your changes (`git commit -am 'Add some feature'`)
4. Push to the branch (`git push origin my-new-feature`)
5. Create a new Pull Request