TP 3 : Javascript + Helpers Ruby On Rails

Objectifs

Utiliser Javascript pour manipuler le DOM et effectuer des requêtes asynchrones. Manipuler quelques helpers Ruby On Rails.

Étapes

Préambule

1. Cloner le repository suivant https://github.com/yannski/tp-blog-rails-exemple dans un dossier qui n'est pas déjà un projet Ruby On Rails :
   `git clone https://github.com/yannski/tp-blog-rails-exemple`
2. Aller dans le dossier tp-blog-rails-exemple qui vient d'être créé et définissez la version 2.3.0 de Ruby :
   `cd tp-blog-rails-exemple`
   `rbenv local 2.3.0`
3. Installer les gems :
   `bundle install`
4. Dans le fichier **`config/database.yml`**, remplacer les valeurs des clés **`username`** et **`password`** pour les environnements **`development`** et **`test`** en fonction du compte PostgreSQL créé durant le TP précédent.
5. Créer les bases de données de l'application, charger le schéma de migration et charger le jeu de test (vérifier qu'une instance de PostgreSQL était bien lancée avant d'exécuter cette ligne de commande) :
   `bundle exec rake db:setup`

Configuration et connexion à la base de données

1. Réaliser les tâches définies sur la page d'index des articles

Il est recommandé de tester le code Javascript dans la console de votre navigateur. Le code Javascript est à écrire dans le fichier **`app/assets/javascripts/base.js`**.

Création du modèle de données

1. Convertir tous les liens de la page d'index des articles en utilisant le helper **`link_to`**
2. Tronquer les articles en utilisant le helper **`truncate`**
3. Créer un formulaire en utilisant le helper **`form_for`** dans la vue correspondante à l'action articles#new pour ajouter un article
4. Ajouter un lien pour chacun des articles de la page articles#index pour supprimer l'article (en base ET visuellement) correspondant de manière asynchrone :
   - en utilisant une requête AJAX avec jQuery
   - en utilisant l'attribut **`remote: true`** et **`jquery-ujs`** (déjà présent dans le projet)