TP : Création d'une plateforme de gestion d'annonces

M2 MBDS

2019-2020

1. Présentation Générale

L'objectif de ce premier TP est de vous aider à mettre en place votre environnement de travail et vous donner toutes les informations nécessaires à la réalisation du projet.

Le développement GRAILS demande une machine d'une puissance respectable puisqu'il vous faudra installer de nombreux éléments demandant une quantité importante de ressources.

Ce projet est à faire en binôme. Un trinôme est accepté en cas de compte impair.

2. Objectifs

a. Installation

Installation des éléments suivants :

- JAVA 8 dans la version correspondante à votre système d'exploitation (32 / 64 bits)
  http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
  Téléchargez et installez
- Grails 3.3.8 ou plus
  https://grails.org/download.html
  Téléchargez et dézippez le dossier à la racine de l'un de vos disques
- IntelliJ IDEA Ultimate (Licence fournie par le MBDS), prenez systématiquement la dernière version disponible sur le site de Jetbrains.
  https://www.jetbrains.com/idea/download/
- Optionnel
  - Un serveur Apache, à installer de préférence en service si vous êtes sous Windows, EasyPHP / WAMP / MAMP par défaut si vous ne savez pas l'installer / le configurer
    https://httpd.apache.org/download.cgi
  - MySQL ou PostgreSQL

Tester votre installation :

- Démarrez IntelliJ
- Créez un nouveau projet
- Sur la gauche vous devriez avoir la possibilité de choisir un projet de type « Grails »
- Fournissez le lien local vers le dossier du SDK Grails que vous avez dézippé plus tôt
- Sélectionnez « create-app »
- Nommez votre application et cliquez sur « Finish »
- Attendez qu'IntelliJ ait fini la création des fichiers de base du projet, qu'il les indexe
- Lancez ensuite le projet afin que votre IDE télécharge toutes les dépendances, cela peut prendre quelques minutes, le bon moment pour une pause !

b. Le sujet

Nous allons maintenant créer le projet sur lequel nous allons travailler pendant les séances de TP :

Nous allons créer une plateforme de gestion d'annonces.

Le rendu de projet prendra trois formes :

- **Un backend permettant d'administrer le contenu**
- **Un frontend simple présentant la liste de toutes les annonces ainsi qu'une page de détail de l'annonce**
- **Une API REST mettant toutes les données à disposition**

c. Contraintes et besoins

Voici la liste des contraintes pour la réalisation de votre projet :

- Vous devrez gérer deux niveaux d'accès via des **Rôles** qui seront attribués à vos **Utilisateurs**.
    - L'administrateur (**ADMIN**) qui aura accès à un backoffice permettant d'opérer ***toutes les fonctions de CRUD*** sur le modèle de donnée existant
    - L'utilisateur (**USER**) qui sera un utilisateur standard ayant comme seule possibilité de soumettre et récupérer des informations sur l'API REST que nous créerons
- Les **Annonces** contiendront toutes les informations relatives aux annonces publiées par les Utilisateurs, ces dernières contiendront :
    - Titre
    - Description
    - Image d'illustration

    - Date de soumission
    - Date de validité
    - Etat (enabled, disabled)
- Les **Utilisateurs** auront quant à eux des informations classiques d'un compte ainsi qu'une **liste d'Annonces :**
    - Nom d'utilisateur
    - Mot de passe
    - Date de création
    - Thumbnail
- De manière générale, lorsque ce sera pertinent, vous garderez trace des dates de création des entités.
- Votre plateforme sera composée de 2 parties
    - Un site qui permettra de créer / modifier / voir / supprimer des **Utilisateurs** et **Annonces**.
    - Une API REST que nous développerons dans un second temps.

Mettez-vous dans la peau du chef de projet qui reçoit la très sommaire expression des besoins décrite plus haut, réfléchissez au modèle de données que vous utiliseriez pour réaliser ce projet.

3. Le projet

a. Création du projet

La création d'un projet Grails demande le respect de certaines étapes, voici un récapitulatif.

- Dans un premier temps vous devez créer votre projet, vous l'avez normalement déjà fait.
- Une fois votre projet créé, faites le tour de fichiers de configuration « grails-app /conf/* » pour voir ce qu'ils contiennent et compléter les éventuelles valeurs à modifier, par exemple dans le fichier « application.yml », de manière générale essayez de comprendre ce qui s'y trouve.
  Vous pouvez ajouter à ce fichier une configuration de base pour le package par défaut dans lequel seront créées vos classes, pour ce faire il suffit d'attribuer la valeur souhaitée sous le chemin « grails.project.groupId », exemple :

```
grails:
    project:
        groupId: 'com.tokidev.whatever'
```

- La pierre angulaire d'un projet Grails réussis, c'est un modèle travaillé et réfléchis, définissez les classes dont vous allez avoir besoin, lancez le projet et observez le résultat, est ce que les tables créées vous permettront de mener votre projet à bien, cette partie est essentielle, prenez votre temps et appelez l'enseignant au besoin pour vérifier.
  Lorsque vous lancez un projet sans définir de base de données, par défaut il sera lancé sur une base de donnée

simulée en mémoire, pour y accéder, il suffit de taper dans votre navigateur « /dbconsole » après la racine de votre projet déployé (par exemple : http://localhost:8080/monprojet/dbconsole.

- Une fois votre modèle complet, vous pouvez passer à l'étape de scaffolding qui va vous permettre de générer une base d'interface, basée sur votre modèle, encore une fois, prudence, les contrôleurs que vous allez générer sont génériques, les vues ne le sont pas, si une fois que vous avez commencé à modifier vos vues, vous modifiez votre modèle, vous devrez régénérer les vues au risque de perdre tout ce que vous avez fait sur ces dernières !

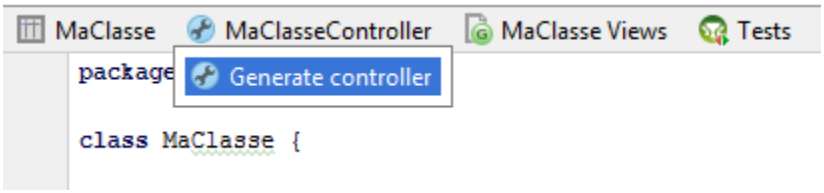

Figure 1: Génération du contrôleur

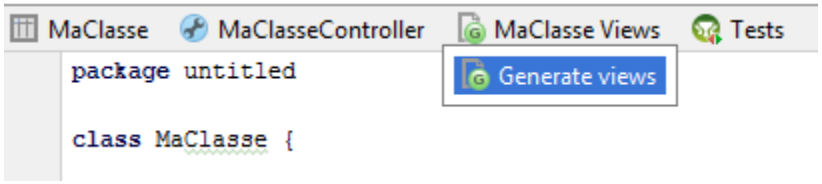

Figure 2: Génération des vues

- Les contrôleurs que vous venez de générer doivent se trouver sous « grails-app/controllers » et les vues dans « grails-app/views » prenez le temps de regarder les fichiers générés pour comprendre le fonctionnement global.
- Vient enfin le moment de définir des données d'initialisation à votre projet, ceci se fait dans le fichier « grails-app/init/**/Bootstrap.groovy » et vous servira à avoir quelques données de test.

b. Utilisation d'une base de donnée type « MySQL » (optionnel)

Je vais vous donner ici les éléments pour accomplir cette tâche.

Si vous préférez utiliser un autre SGBD, libre à vous !

1) Installer MySQL
2) Suivre les indications de la documentation : http://docs.grails.org/latest/guide/conf.html#dataSource
3) **Créez votre base de données**, Grails crée les tables mais pas la base.
4) Relancez votre projet. Si vous voyez vos tables se créer, tout est bon !

c. Ajout de données

Modifiez le « BootStrap.groovy » afin d'avoir au lancement de votre projet au moins :

- Un utilisateur pour chaque rôle
- Quelques entrées pour chacune des autres entités

4. Informations diverses

- Construction du backend d'administration, vous avez deux possibilités :
    - Générer un backend avec les options de scaffolding et modifier les vues / controlleurs
    - Partir de rien et tout faire à la main
- Elle devra évidemment permettre de créer / modifier toutes les entités existantes tel que décrit dans la partie « Contraintes et besoins »
- La visualisation des images devra se faire sur les différentes pages, simplement afficher le chemin de l'image n'est pas considéré comme une solution viable.
- Les formulaires de modification et de création impliquant des images devront permettre l'upload de ces dernières.

5. Bonus / Malus

Ces points vous apporteront des bonus lors de la notation :

- Pour les parties relatives à l'ajout / modification de fichiers image, mettez en place sans utiliser d'extensions, la possibilité d'uploader les fichiers en Ajax en faisant un simple drag'n'drop du fichier sur le champ en question.
- Mettez en place un cron fonctionnel pour désactiver les annonces lorsqu'elles passent leur date de validité.

Un gros malus sera attribué aux étudiants qui ne chiffreront pas les mots de passes des utilisateurs

6. Informations complémentaires

Afin de stocker les images que vous allez uploader via les différents formulaires, vous devrez les enregistrer mais Grails ne mets pas par défaut à disposition d'espace de stockage de fichier persistant, si vous stockez les fichiers dans un répertoire public de votre projet Grails, ceux-ci seront perdu au redémarrage de votre serveur d'application.
Deux méthodes viables :

1. Externaliser le répertoire de stockage des assets (Slide 60 du cours)
2. Installer un serveur Apache et y stocker les fichiers

Votre projet référencera le chemin interne et externe des images (path & url), la meilleure façon de faire reste de stocker uniquement le nom / la fin du chemin vers le fichier en base de données et d'avoir la base du path et de l'url dans un fichier de configuration pour pouvoir reconstruire le path et l'url à la volée.

7. Rendu

Ce projet sera noté à la fin des séances et constituera une part importante de votre notation dans ce module (moitié de votre note)

La date limite de rendu est fixée au vendredi 4 octobre si l'emploi du temps ne subis pas de modifications.
Les projets seront présentés lors du dernier cours, groupe par groupe.

La soumission des sources se fera obligatoirement via un **git publique** sur lequel vous aurez **mis à jour régulièrement votre projet au plus tard le dimanche 6 octobre**.

Vous devrez envoyer un mail à mon adresse (greg.galli@tokidev.fr), le **sujet du mail** sera formaté de la manière suivante : [MBDS_GRAILS_REST] Rendu projet – « Votre Nom » (sans les doubles quotes évidement), vos mails seront triés par un filtre, si vous ne respectez pas ceci, je ne verrai pas votre mail.

Votre git devra contenir un **readme** précisant certains éléments que vous souhaiteriez éclaircir.

Si vous avez des questions, adressez-moi un mail à l'adresse greg.galli@tokidev.fr, je ferai de mon mieux pour revenir vers vous le plus rapidement possible.

Des bonus seront accordés aux groupes qui feront le choix d'implémenter les « Bonus » / à ceux qui s'appliqueront particulièrement sur la propreté du code.

Un design soigné et une bonne ergonomie sont des plus qui seront pris en compte dans la notation de votre projet sans l'impacter de manière dramatique.

Vous serez noté en tant que binôme sauf si j'estime qu'un élève a fournis beaucoup plus de travail que l'autre, auquel cas, les deux étudiants pourront avoir des notes différentes.

Les projets copiés se verront systématiquement attribuer une note de 0.

Attention : Si vous avez utilisé une base MySQL ou autre pour votre projet, vous penserez avant le rendu à repasser sur la base de donnée mémoire initialement configurée, je ne peux pas me permettre de créer une base pour chacun des groupes, si vous oubliez cette partie, un malus sera appliqué.