Master 1 informatique – Université Claude Bernard Lyon 1 – mardi 8 décembre 2009

MIF 13 - Programmation Web

Durée : 2 heures – Documents autorisés – Calculatrices et téléphones portables interdits

Questions de cours (barème : 8 points)

1. A l'heure actuelle, pour faire une application géolocalisée sur téléphone mobile, concevriez-vous plutôt une RIA (Rich Internet Application) ou une RDA (Rich Desktop Application) ? Justifiez votre réponse.

   Actuellement, il n'y a pas de navigateur qui prenne en compte la recommandation HTML 5 sur la géolocalisation. Il faut donc utiliser un client lourd ➔ RDA

2. Quels sont les intérêts et les inconvénients d'utiliser SVG (Scalable Vector Graphics) dans une application Web de type RIA ?

   Intérêt : syntaxe XML, donc générable / utilisable par tous les outils et technos XML (par exemple, AJAX).
   Inconvénient (majeur) : pas supporté par tous les navigateurs…

3. Vous souhaitez transformer une application Java standalone en RIA. Utiliseriez-vous plutôt une bibliothèque directe (JQuery) ou indirecte (GWT) ? Justifiez votre réponse.

   Une bibliothèque indirecte comme GWT permet de réaliser des applications quasiment standalone côté client qui s'approcheront plus du modèle actuel de votre application (cf. tortues).

4. Pourquoi un framework Web MVC n'est-il pas incompatible avec une bibliothèque AJAX indirecte ? Quel serait le résultat du code produit par l'association des deux ?

   Rien n'empêche en théorie Struts et GWT par exemple de fonctionner ensemble. L'un permet d'organiser son code, et l'autre de générer, une fois le code écrit, du JavaScript qui sera déployé sur le client. Il existe d'ailleurs des implémentations de frameworks Struts + GWT qu'on peut trouver sur le Web. Le résultat est du code JavaScript structuré en MVC.

Etude de cas (barème : 12 points)

La Ville de Lyon a décidé de mettre en place un système d'information lié à l'évaluation des animations de la fête des Lumières (LUMEVAL). Les visiteurs de la fête devront ainsi pouvoir commenter, et échanger sur les animations, soit pendant leur déambulation, soit plus tard. La Ville de Lyon compte aussi sur le système pour évaluer comment améliorer les choses d'une année sur l'autre.

Vous avez déjà déroulé plusieurs phases du processus de conception. Les choix de conception sont les suivants : un framework MVC (Struts) et un ORM (Hibernate).
Vous en êtes à la réalisation des itérations, et en particulier à celle du cas d'utilisation suivant :

Nom : commenter une animation sur le Web
Contexte d'utilisation : une personne ayant assisté à la fête des lumières et ayant un commentaire à faire sur une animation l'entre dans le système. Le commentaire pourra être publié par un animateur sur le site ou uniquement archivé afin de permettre un traitement statistique ultérieur.
Portée : système boîte blanche
Niveau : objectif utilisateur
Acteur principal : utilisateur anonyme
Intervenants et intérêts :

- animateur du site, qui souhaite publier les témoignages les plus éloquents et conserver une trace des opinions des autres utilisateurs

Pré-conditions :
- les animations ont été répertoriées et leurs noms sont disponibles dans le système
- la date et l'heure de la cérémonie d'ouverture de la fête des lumières sont passées
- l'utilisateur a accédé au formulaire de saisie de commentaires

Garanties minimales : aucune

Garanties en cas de succès :
- le commentaire saisi par l'utilisateur est persisté et accessible par l'animateur
- l'utilisateur est informé que son commentaire a bien été enregistré et sera traité dans les plus brefs délais

Déclencheur : un individu assiste à une animation et souhaite faire un commentaire à son propos

Scénario nominal
1. L'utilisateur sélectionne l'animation sur laquelle porte son commentaire
2. Il tape son commentaire
3. Il tape le texte contenu dans le pictogramme
4. Il valide le formulaire
5. Le commentaire en attente est transmis au serveur qui l'enregistre en base
6. Le système renvoie à l'utilisateur une page lui indiquant qu'il a bien enregistré son commentaire et le lui affichant

Extensions

4.a : Le texte de vérification ne correspond pas à celui du pictogramme

4.a.1 : le système renvoie la page de commentaire avec un nouveau pictogramme et un message demandant à l'utilisateur de taper le nouveau texte.

4.b : Panne de réseau

4.b.1 : le client affiche un message d'alerte lui conseillant de copier son message et de le sauvegarder pour pouvoir le renvoyer plus tard.

5.a : Erreur d'enregistrement dans la base de données

5.a.1 : le serveur renvoie une page d'erreur 500

Informations connexes : on dispose d'un sous-système générant des pictogrammes contenant un texte et capable de valider si ce texte correspond bien à une chaîne passée en paramètre, afin d'éviter les attaques par saisie automatique qui surchargeraient le site.

Structure de l'application (barème : 8 points)

5. Listez les différents éléments (objets, fichiers de configuration…) mis en œuvre dans ce CU, en fonction de l'architecture choisie. On ne s'intéresse pas ici à la base de données.

   Variable selon vos réponses : 2 JSP (celle ci-dessous) + une qui informe que le commentaire a été validé ; les 2 beans donnés dans l'énoncé ; Action, ActionForm, fichier Web.xml, Struts-config.xml, ApplicationResources.properties.

6. Ecrivez un diagramme de séquence représentant les communications entre les différents objets de votre application pour réaliser le scénario nominal (sans extension)

   …

7. Proposez une solution technique générale et donnez l'algorithme correspondant pour réaliser le comportement décrit dans l'extension 4.b.

   Utiliser la fonction de validation de formulaires AJAX pour envoyer une requête asynchrone au serveur. Si la réponse est obtenue, valider le formulaire. Sinon, afficher une boîte d'alerte et renvoyer false pour que le formulaire ne soit pas envoyé au serveur.

8. Ecrivez la partie du fichier Struts-config.xml permettant de faire fonctionner les éléments que vous avez listés.

```
<form-beans>
  <form-bean name="PostActionForm" type="org.lumieres.commentaires.PostActionForm" />
</form-beans>
<action-mappings>
  <action input="/CommentaireForm.jsp"
    name="PostActionForm"
    path="/commentaire"
    scope="session"
    type="org.lumieres.commentaires.NewStrutsAction" />
  <action path="/Comment" forward="/CommentaireForm.jsp" />
  <action path="/valideComment" type=" org.lumieres.commentaires.PostAction" name="valide">
        <forward name="succes" path="/ValideOK.jsp"/>
        <forward name="echec" path="/CommentaireForm.jsp?message=Erreur+de+validation"/>
  </action>
</action-mappings>
```

Indications :

Pour réaliser ce CU, vous disposez de deux beans :

- Un bean « *Pictogramme* », exposant les propriétés suivantes :
  - *String URLPicto* : l'URL vers un pictogramme contenant un texte à taper (accessible en lecture uniquement) ; chaque lecture provoque la création d'un nouveau pictogramme
  - *String Text* : texte à valider par rapport au pictogramme dont l'URL a été donnée (accessible en écriture uniquement)
  - *boolean valid* : true si le texte correspond au pictogramme, false sinon (accessible en lecture uniquement)
- Un bean « *Commentaire* » (destinée à la persistance des commentaires), exposant les propriétés suivantes :
  - *int animationNumber* : le numéro de l'animation sur laquelle porte le commentaire
  - *String text* : le texte du commentaire

Vous utiliserez la même JSP pour la saisie initiale du commentaire et pour son renvoi en cas d'erreur dans la saisie du texte de validation. Bien entendu, dans ce dernier cas, l'utilisateur retrouvera la sélection de l'animation et le texte de son commentaire, plus un message lui indiquant que le texte saisi ne correspondait pas et qu'il est invité à recommencer.

Programmation (barème : 4 points)

Ecrivez le code de la JSP ci-dessus.

…