IFT 1148 Programmation ASP.NET

Hiver 2018 **par Alain Pilon**

Travail pratique #3 (version préliminaire)

Date de remise

Lundi 16 avril à 23h55 (les retards seront pénalisés de 20% par jour - la note de 0% sera attribuée après plus de 3 jours de retard)

Attention n'attendez pas à la dernière minute, ce travail est une bonne pratique pour l'examen final.

Remise électronique dans Studium comme pour les TP1 et TP2

Barème

Ce travail compte pour 15% de la note finale du cours.

À faire en équipe de deux de préférence (ou individuellement), inscrivez clairement les noms des équipiers dans chacun des fichiers. Un des équipier remet le travail et le second remet le fichier RemiseCoequipier.

Objectifs

Le but principal de ce travail est de vous permettre de

- ajouter,
- consulter,
- supprimer et
- modifier le contenu d'une base de données à partir d'une page ASP.NET.

Plus spécifiquement, vous aurez à utiliser les contrôles Web

- de gestion des bases de données comme les
  - DataSources,
  - GridView et
  - DetailsView.
- vous utiliserez également les
  - TemplateFields des GridView,
  - la validation des objets ASP.NET lors de l'édition des données d'un GridView,
  - l'ajout de contenus dans des colonnes par programmation dans un GridView.
- vous aurez à utiliser Ajax et plus spécifiquement
  - ScriptManager,
  - UpdatePanel et
  - UpdateProgress.

Travail à faire

Pour ce travail vous allez utiliser la base de données `notes.mdb`.

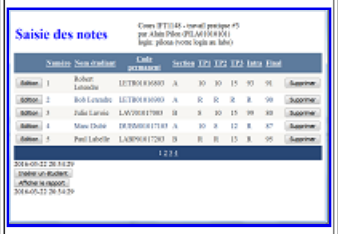
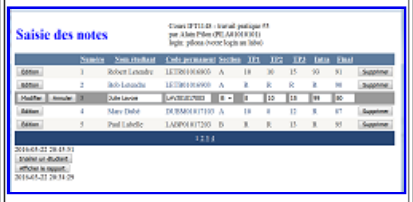
- La structure (dictionnaire de données) est déjà présente et quelques étudiants ont déjà été insérés.
- Vous remarquerez que les notes sont toutes valides: les données sont validées lors de la saisie et n'ont plus à être validé par la suite.
- Pour faire vos tests vous pouvez ajouter d'autres étudiants, mais n'enlevez pas ceux qui s'y trouvent déjà.

Le travail consiste à fournir une interface pour récupérer, ajouter, modifier, ou supprimer les informations d'étudiants (nom, code permanent, notes, etc.) à l'aide d'une base de données. L'utilisateur aura ainsi la possibilité de consulter les fiches des étudiants, d'en ajouter de nouvelles, de les modifier, ou de les supprimer.

- La suppression d'une fiche devra être confirmé par une fonction JavaScript permettant d'annuler l'opération en cas d'erreur (comme on l'a fait pendant le cours).
- L'ajout d'un nouvel étudiant doit se faire avec un DetailView (invisible initialement) et d'un bouton "Ajouter un nouvel étudiant" (comme dans le cours).

    - Ainsi initialement (pour l'ajout) on affiche le GridView et le bouton "Ajouter"
    - Lorsque l'utilisateur clique sur le bouton, le GridView et le bouton disparaissent pour faire place au DetailView (déjà en mode d'insertion)
    - Après que l'utilisateur confirme ou annule l'ajout, le DetailView est remplacé par le GridView et le bouton.
- La modification des fiches se fera directement dans le GridView où pour chaque champ vous devrez utiliser des contrôles Web de validation.
    - La section est choisie dans une liste déroulante construite à partir des valeurs (uniques) déjà présentes dans la table;
    - Pour éviter que le GridView soit trop large lors de l'édition des données j'ai changé la largeur de tous les "Textbox";
    - J'ai utilisé la propriété "column" et placé les valeurs:
        - 15 pour les nom et code permanent;
        - 5 pour toutes les notes.
- L'ajout d'une fiche se fait dans un DetailView où vous devez inclure tous les contrôles Web de validation (voir plus loin). De plus vous devez fournir une liste déroulante pour la sélection de la section (comme montré à la figure 4).
- Les validations suivantes devront être faites (lors de l'ajout et des modifications):
    - Le nom sera validé en utilisant un RFV (RequiredFieldValidator) uniquement (pas d'autre validation nécessaire);
    - Le code permanent sera validé en utilisant un RFV et une validation par expression régulière (4 lettres suivies de 8 chiffres);
    - Comme pour le TP2 les notes doivent être validées en utilisant un CustomValidator;
    - Les notes valides sont "R" ou une note entière entre 0 et
        - 10 pour les TP1 et TP2;
        - 20 pour le TP3;
        - 100 pour intra et final.
        - Pour des raisons de simplicité au niveau de l'affichage, les notes saisies seront toutes des valeurs entières.
        - Attention la pondération a changée par rapport au TP2.

Tout se fait sur le même écran. Vous devez reproduire avec le plus d'exactitude possible la page ASP.NET suivante: (cliquez sur les images).

Figure 1. Affichage initial	Figure 2. Modification d'un étudiant	Figure 3. Affichage du rapport	Figure 4. Ajout d'un étudiant
			

La première rangée de la page Web affiche le titre en h1 "Saisie des notes" en bleu. À sa droite vous mentionnez vos renseignements personnels pour vous identifier auprès du correcteur (si vous avez fait le travail à deux n'oubliez pas de mettre vos deux noms).

Dans le `GridView` du haut, qui affiche les informations des étudiants,

- vous chargez tous les champs de la base de données en mode lecture/écriture, sauf pour le numéro de l'étudiant qui sera en lecture seulement.
- Activez le tri et la pagination à raison de 5 étudiants par page dans votre GridView.
- Dans la première colonne vous insérez des **boutons** d'édition (*Edit, Update, Cancel*).
- Dans la dernière colonne vous insérez un bouton de suppression (pour chaque rangée). La suppression se fera seulement suite à une confirmation JavaScript.
- Les mises à jour devront bien entendu être reproduites dans la base de données au fur et à mesure des modifications.
- Changez le texte des boutons pour afficher uniquement en français.
- Changez le texte des entêtes pour afficher comme dans mon exemple.

Immédiatement en dessous du GridView il faut affiché le texte "Affiché le:" et l'heure du système. Vous placez un "Label" et l'heure est insérée sur l'événement de **chargement de la page**.

Le rapport:

- Le bouton pour afficher le rapport des notes fera afficher toutes les informations de la base de données **sans valider son contenu**.
- **La note finale du cours pour chaque étudiant** est calculée et affichée à la droite de ses notes détaillées.
- Vous pouvez utiliser les fonctions de votre TP2 pour calculer la note finale, par contre si vous avez des Sub ou des Function qui dépassent 40 lignes vous devrez les découper pour qu'ils soient plus faciles à lire. Attention la pondération a changée par rapport au TP2.

- Affichez le nom, code permanent, toutes les notes provenant de la base de données et la note finale (comme pour le TP2 en tenant compte des R).
- Si la note finale est inférieure à 50% vous ajoutez le message "échec" en rouge.
- Je vous suggère de faire comme moi et d'utiliser un GridView (j'ai ajouté deux colonnes, une pour la note et l'autre pour le mot échec lorsque nécessaire) pour produire ce rapport, mais vous pouvez utiliser le contrôle Web que vous voulez (label, ListView, Table, Repeater, etc) à ce moment affichez un étudiant par ligne et assurez-vous que ce soit lisible et bien présenté.
- Immédiatement en dessous du rapport il faut ajouter un label qui sera affiché en même temps que le rapport. Ce label contient le message "Affiché le: " et l'heure du système. L'heure doit y être inséré sur l'événement de **chargement de la page**.

Ajax:

- Insérez deux "UpdatePanel" sur votre page ASP.NET
- Dans le premier, placez-y le GridView pour la mise-à-jour de la BD, ainsi que le bouton "ajouter", le DetailsView d'insertion et le label qui affiche l'heure.
- Dans le second, placez-y le bouton "Afficher le rapport" avec le GridView (ou autre contrôle si vous en avez utilisé un autre) pour le rapport et le label qui affiche l'heure.
- Ajoutez au tout début de la page un "UpdateProgress" qui sera affiché après 1 seconde d'attente jusqu'à ce que la page soit rechargée lors d'un rechargement partiel provoqué par le second "UpdatePanel" (celui pour le rapport). Le texte doit affiché dans une balise h1 en bleu "Patientez, construction du rapport...".
- Ajoutez un délai de 3 secondes (comme montré dans le cours en utilisant System.Threading.Thread.Sleep), pour simuler un délai sur l'événement "chargement de la page".
- Ainsi vous devriez avoir l'affichage de l'heure à 2 endroits sur votre page. Lors du chargement initial de la page vous devriez avoir la même heure, mais par la suite un changement dans le premier GridView devrait provoquer la modification de la première heure alors qu'un clique sur le bouton "Afficher le rapport" devrait réafficher le rapport et modifier la deuxième heure.

Je vais faire une courte démonstration pendant le cours du 19 mars.

Notes:

- l'événement du chargement de la page devrait contenir:
  - le délai de 3 secondes (System.Threading.Thread.Sleep(3000)) et
  - l'affectation de l'heure (DateTime.Now) dans les deux labels.

Évaluation:

- Fonctionnement et exactitude des résultats:
  - Gestion des données de la base de données (lecture/modification/effacement/ajout): 5 pts
  - Affichage du rapport: 2 pts
- Création et utilisation de fonctions et de sous-routines: 2 pts. *Lorsqu'un bloc d'instructions est complexe ou long, découpez le en plusieurs sous-blocs (sous-routines)*. Un bloc d'instructions qui dépasse 40 lignes (excluant les commentaires) sera pénalisé:
  - Si vous avez un bloc d'instructions qui dépasse 40 lignes (excluant les commentaires) vous aurez 1 point sur 2;
  - Si vous avez un bloc d'instructions qui dépasse 60 lignes vous aurez 0 point sur 2;
  - Si vous avez plusieurs blocs d'instructions qui dépassent 40 lignes vous aurez 0 point sur 2.
- Présentation du code ASP.NET et script du côté serveur (avec commentaires pertinents): 1 pt
- Validation des valeurs entrées: 2 pts
- Calcul des valeurs du rapport (notes dans l'étendue de valeurs, code R, échec du cours): 2 pts
- Utilisation de Ajax: 1 pt