Documentation technique

# View

## Sélection de la chaine de connexion (dans les propriétés de l’appli)

Il est possible pour de multiples développeurs de compiler le code source en mode *debug* et alors de choisir sa chaîne de connexion via une liste déroulante préalablement remplie avec leurs chaînes commençant par « JobOverviewConnectionString ».

En mode *release* la chaîne  « JobOverviewConnectionStringDefault » est utilisée.

A gérer : avoir un fichier de configuration différent si la compilation se fait en *debug* ou en release.

Pour la gestion des droits, la solution adoptée est la gestion des commandes en fonction d’un mode qui est un énumérable, ce qui permet une meilleure évolutivité en cas d’ajout de nouveaux types d’utilisateur.

Fenêtre modale pour création de taches avec champs pré remplis mais modifiables : plus cohérent avec une insertion

Pour la saisie des temps, on utilise un *expander* qui semble plus cohérent qu’une fenêtre modale dans le cadre de modification.

Pour l’ajout des taches, on utilise une fenêtre modale avec des *comboboxs*, ce qui rend l’ajout plus rapide.

# Model

## UpdateDatabaseWorkTimeOfTaskList

Utilisation d’un *merge* pour faire l’ajout, la suppression et la modification en une seule requête/connexion et permettant une insertion de masse.

## UpdateDatabaseTaskListOfEmployee

Séparation des requêtes d’ajout et de suppression de taches en masse car celles-ci nécessitent de travailler sur deux tables différentes (jo.Tache, jo.TacheProd) empêchant le *merge*.

Le choix de faire l’ajout et la suppression dans la même méthode permet de n’utiliser qu’une connexion et de s’assurer que le tout ne sera effectué que si aucune erreur ne se produit (requête effectuée dans une transaction).

## ExportListTaskEmployeeToXML

Utilisation d’un *XMLSerializer* car l’architecture de notre liste d’employés correspond exactement à celle du fichier XML voulu.

# ViewModel

## VMTaskConsultation

### ListTaskToAddOrDelete

Le choix de ne créer qu’une liste pour les insertions, les modifications et les suppressions permet de faire appel à une seule méthode de DAL, il fallait pouvoir distinguer les modifications des suppressions et pour cela la propriété **Productivity** du temps de travail est passée à -1 dans le cas d’une suppression (La productivité est évaluée de 0 à 1).

### Mode édition

Une propriété **EditionMode** de type énumérable permet d’activer et de désactiver les différentes commandes de cette classe, le mode par défaut étant *consultation*.

### AddWorkTime

Cette méthode permet de créer un nouveau temps de travail et l’affecter au temps de travail courant et de passer en mode *édition*.

### ModifyWorkTime

Cette méthode permet de passé en mode **modify**.

### DeleteWorkTime

Ajout du temps de travail courant à une nouvelle liste de temps de travail qui sera elle-même ajouter à la liste **ListTaskToAddOrDelete** comme propriété d’une nouvelle tâche.

L’Id de cette nouvelle tâche sera égal à l’Id de la tache courante.

### ValidateWorkTime

Cette méthode permet de faire la vérification de la validité des données rentrées par l’utilisateur (temps de travail avec date unique pour chaque tache et temps maximum de travail total sur une journée de 8h).

Si les données sont valides, on les inserts dans la liste **ListTaskToAddOrDelete** de la même manière qu’avec la suppression et repasse en mode **consultation**.

Si les données ne sont pas valides on reste dans le mode actuel et on affiche un message d’erreur pour signaler la non validité des données.

### CancelWorkTime

Cette méthode permet de repasser en mode **consultation** et affecte le premier temps de travail de la tache courante au temps de travail courant.

### SaveModification

Cette méthode permet d’appeler la méthode model pour insérer la liste **ListTaskToAddOrDelete** dans la base de données.

On réinitialise ensuite la liste **ListTaskToAddOrDelete.**

## VMTaskManaging

### Propriété SelectedEmployee

Il s’agit de l’employé sélectionné dans la première liste. Cette propriété permet de charger la liste de ses tâches seulement si elles sont consultées. Les listes permettant l’affichage sont alors misent à jour (**ListTaskProd** et **ListTaskAnnex**) ainsi que les autres données relative à l’employé sélectionné.

### Listes d’ajout et suppression de tâches

L’ajout et la suppression de tâche se fait via deux listes l’une gérant les tâches à ajouter et l’autre les tâches à supprimer pour faciliter les interactions avec le model et la fenêtre modale d’ajout de tâche.

### Fenêtre modale de création de tâches VMAddTask

L’employé courant de la fenêtre de création de tâche est envoyé en variable pour **VMAddTask** permettant ainsi de faire le lien entre les deux vues.

Le contrôle d’insertion de valeurs lors de la création d’une tâche se fait via des listes déroulantes pour les activités, logiciels, versions et modules. Pour la saisie de temps prévue pour la tâche le contrôle est effectué via la propriété liée (*Binding*) à la *TextBox*, il est ainsi possible de vérifier la taille du champ renseigné ainsi que le type de donné (si l’utilisateur entre une lettre, une boîte de dialogue est générée et l’avertit et l’ajout de la lettre est annulé).Pour les Label et Description aucun contrôle n’est effectué.