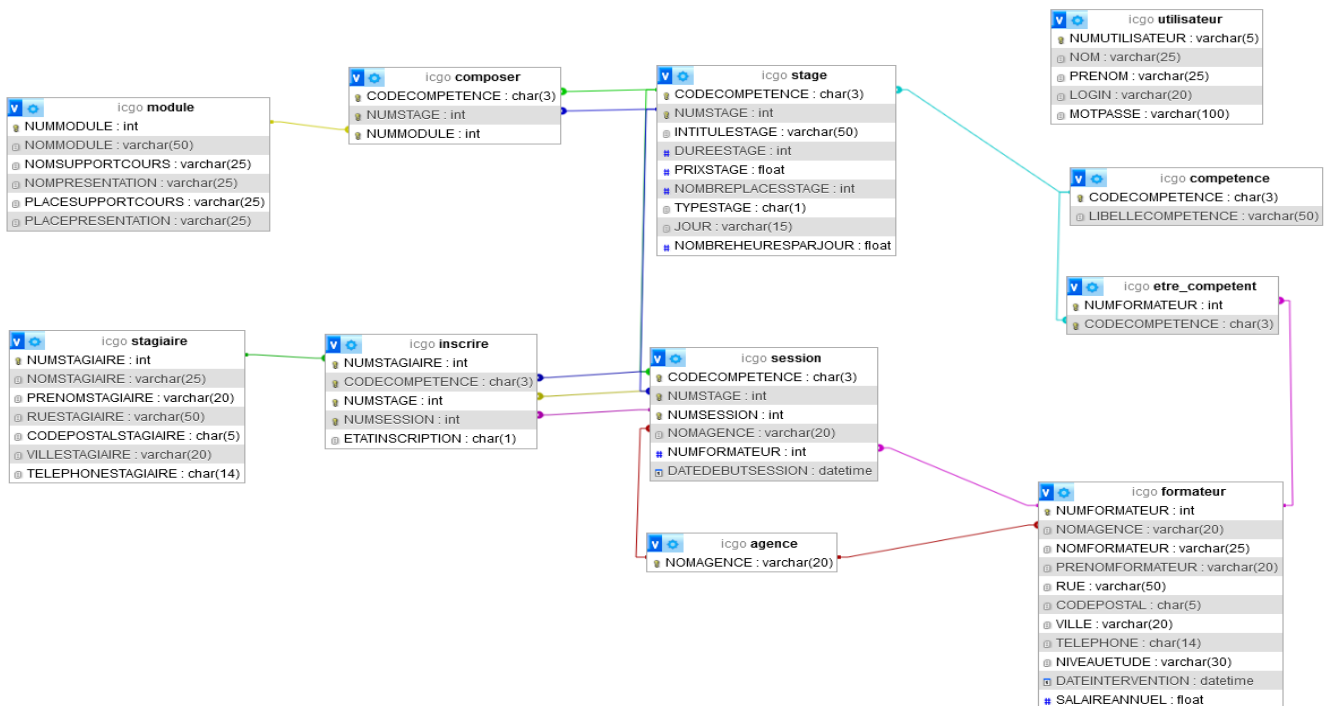
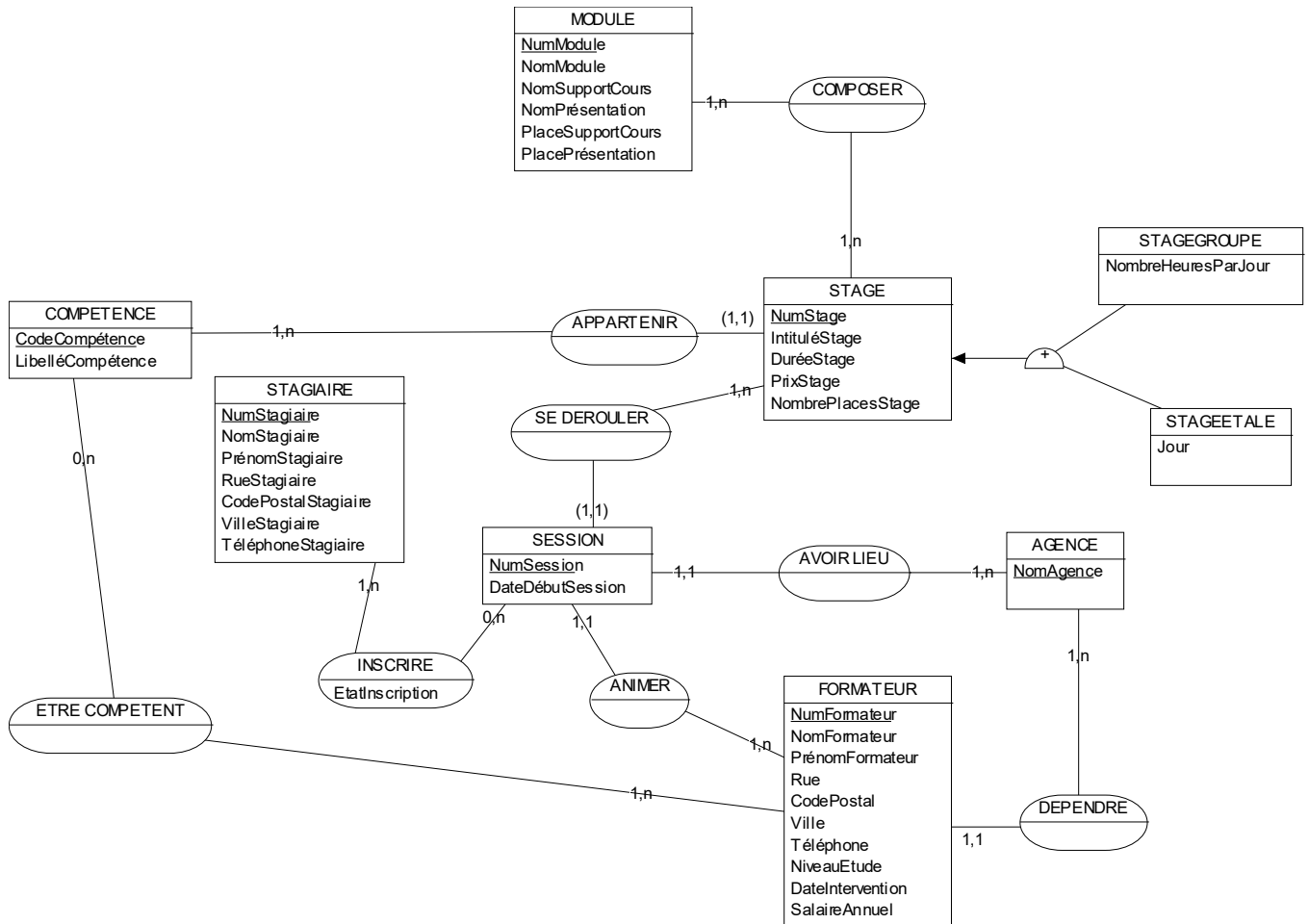


Modèle de données



Les formateurs ont des compétences (pour assurer une partie des enseignements proposés) et dépendent d'une agence (mais peuvent assurer des sessions de formation dans d'autres agences) et chaque formateur anime l'intégralité des cours d'une session de stage.

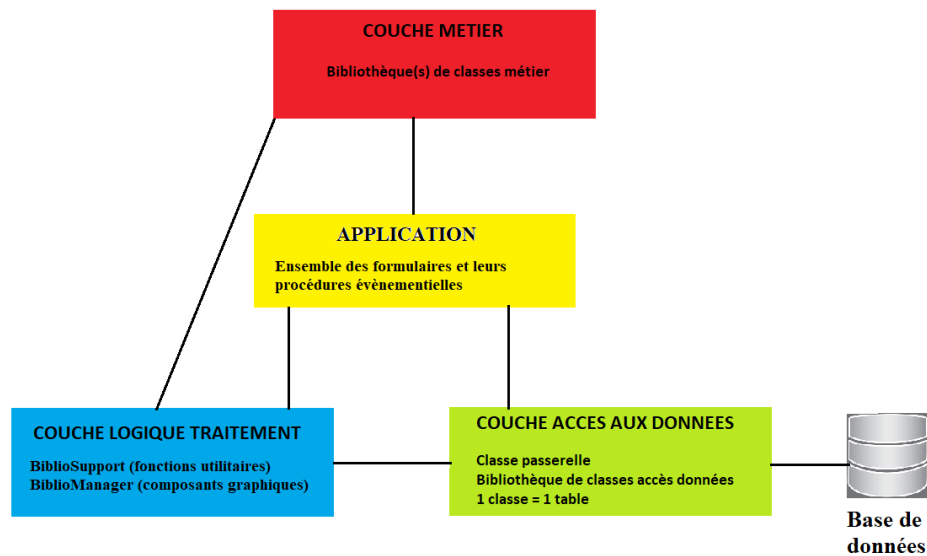
Les stagiaires font des demandes d'inscription à des sessions de stage.

Les stages (groupés -jours se suivent- ou étalés -tous les le jour semaine pendant x semaines-) sont organisés en session pour permettre d'assurer à plusieurs reprise un même stage s'il est très prisé voire dans les différentes agences. Certains enseignements sont communs à plusieurs stages, aussi un stage est composé de plusieurs modules et chaque module peut-être commun à plusieurs stages.

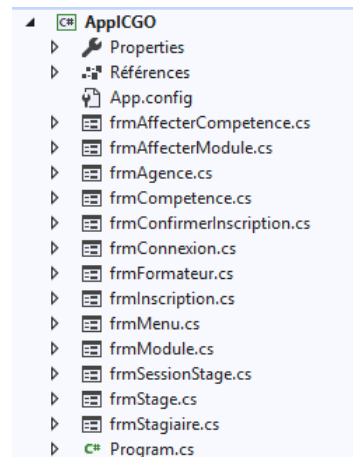
La gestion des inscriptions se fait par courrier. Si la demande d'inscription intervient après la date de début ou si le nombre d'inscrits est dépassé, alors un courrier est envoyé et l'inscription n'a pas lieu. En cas d'une demande incomplète (pas autorisation etp) l'inscription est placée à **P**rovisoire en attente de l'autorisation. Quand l'autorisation est prévue, alors l'inscription passe à **D**éfinitif.

Pour permettre la connexion à l'application, une table Utilisateur recense les personnes qui ont le droit de se connecter.

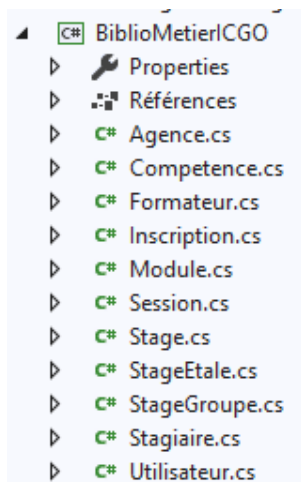
Architecture de l'application



La solution

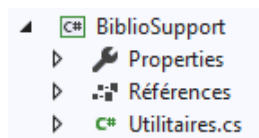


L'application gérée via des Windows Forms (WF)



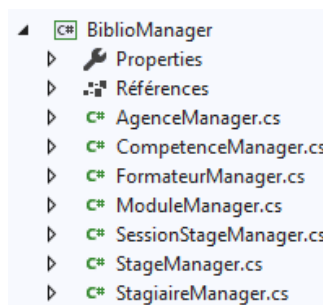
Bibliothèque des classes métiers gérées par AppICGO
A partir de **diagramme de classes**

Couche métier

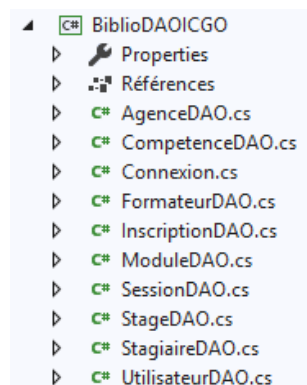


Bibliothèque qui contient une seule classe organisée en régions qui regroupe des fonctions utiles dans les différentes WF permettant d'extraire les identifiants à partir d'un libellé choisi dans un comboBox

Couche logique de traitement



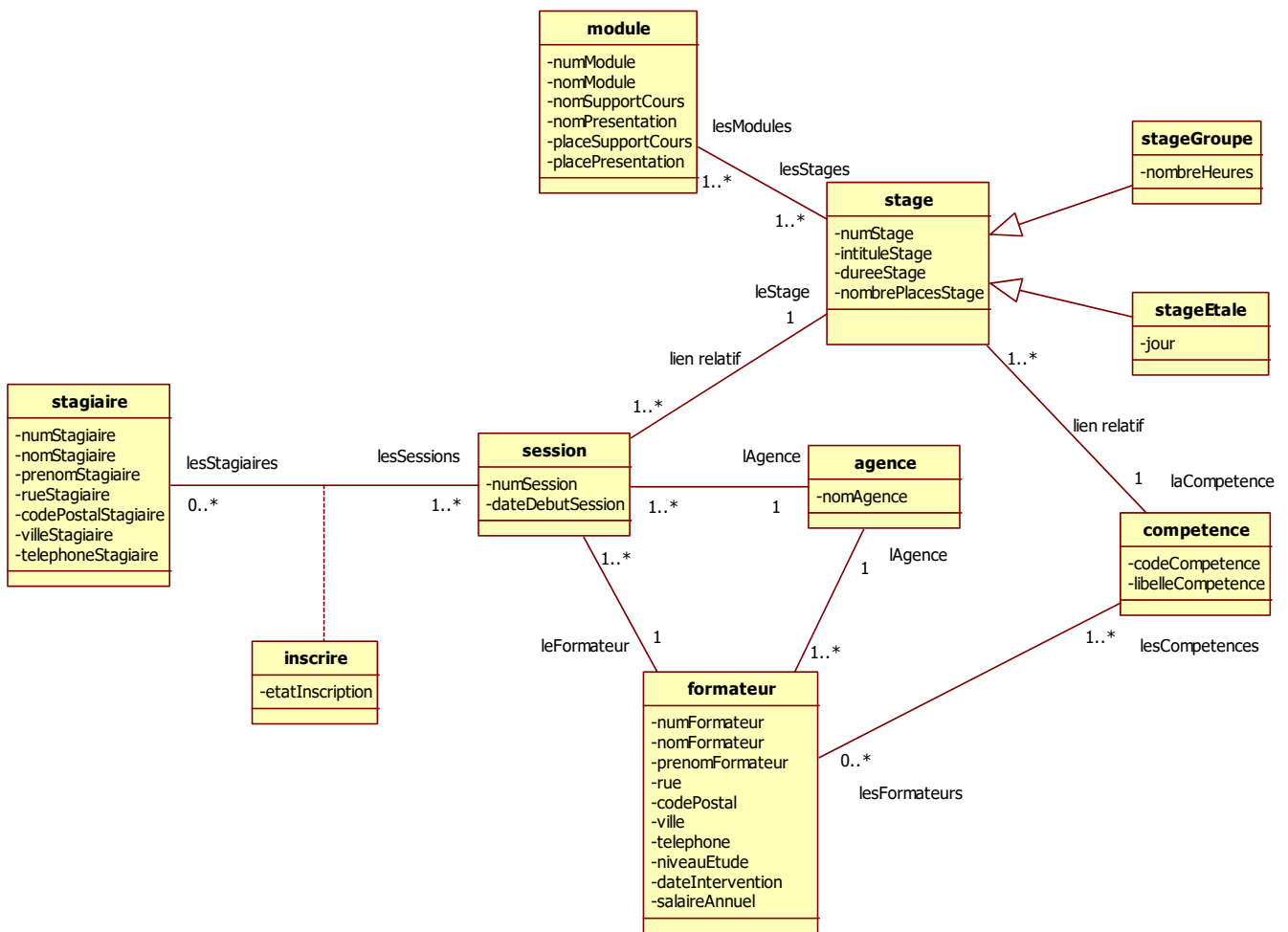
Bibliothèque de classes qui contiennent des fonctions pour gérer les composants graphiques dans les différentes WF (charger les infos de la BD dans les comboBox (organisation par classes et non par région comme BiblioSupport))



Toutes les interactions avec la BD (CRUD)
A partir du **MCD|MPD**
1 classe par table
+ classe connexion

Couche accès aux données

Diagramme de classes



Attention : Il y a une classe inscription avec leStagiaire, laSession et etatInscription.

Il faut se référer à ce diagramme pour gérer la Bibliothèque de classes métiers BiblioMetierICGO et compléter les classes non gérées.

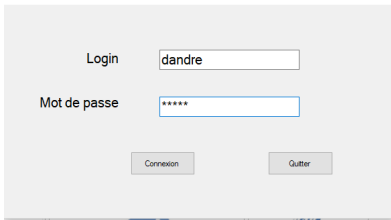
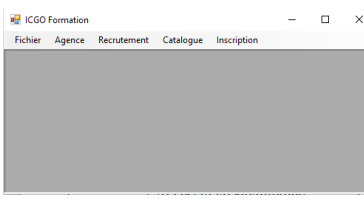
Test application

Cas d'utilisation « Se connecter »

Nom cas d'utilisation : Se connecter
Acteur déclencheur : Administratif
Pré conditions : Néant
Post conditions : L'utilisateur est reconnu administratif
Scénario nominal : <ul style="list-style-type: none">• 1- Le système affiche un formulaire de connexion• 2- L'utilisateur saisit son login et son mot de passe et valide• 3- Le système contrôle les informations de connexion et affiche le menu de l'application.
Exceptions : <ul style="list-style-type: none">• 3-a : le nom et/ou le mot de passe n'est pas valide<ul style="list-style-type: none">3-a.1 Le système en informe l'utilisateur ; retour à l'étape 1• 4- L'utilisateur quitte l'application (manque la confirmation)

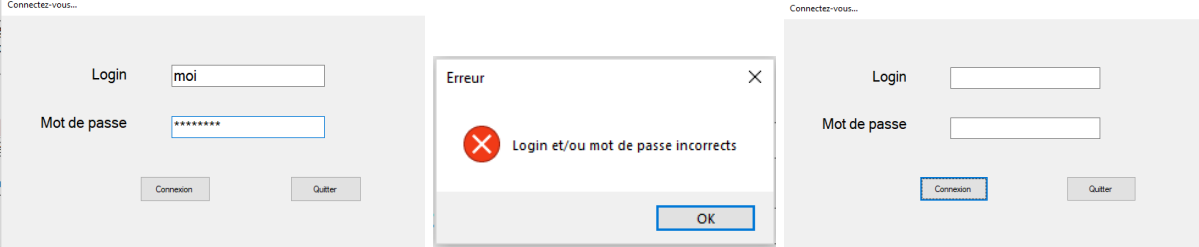
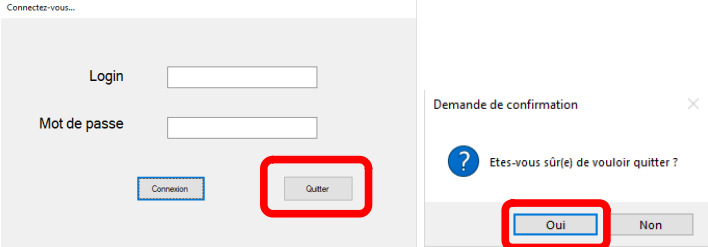
Scénario optimiste : l'utilisateur existe bien dans la table utilisateur.

login «**dandre**», mot de passe «**oppg5**»

Étapes du cas testées	Description du test effectué	Résultat attendu	Résultat obtenu
1	Au lancement de l'application accès à la page de connexion	Affichage de la page de connexion de l'application.	Affichage de la page de connexion de l'application.
2, 3	Entrée du login et du mot de passe valides dans les zones de texte correspondantes. login dandre pwd oppg5	Connexion puis affichage du menu de l'application.	Connexion puis affichage du menu de l'application
			

Scénario pessimiste : le login et/ou le mot de passe sont erronés

login «**moi**», mot de passe «**cbienmoi**»

2, 3-a	Entrée d'un login et d'un mot de passe invalides dans les zones de texte. login moi pwd cbienmoi	Le système informe l'utilisateur login et/ou mot de passe incorrects Retour au formulaire de connexion	Affichage d'un popup d'erreur et retour à la page de connexion.
			
4	Demande de déconnexion suite à un clic sur le bouton Quitter.	Fermeture de l'application	Popup de confirmation. Clic sur le bouton Oui. Fermeture de l'application.
			

Les étapes correspondent aux numéros des échanges utilisateur/système présents dans la description du cas d'utilisation, il peut y avoir plusieurs étapes, séparées par une virgule, par ligne.

Faire ceci pour chaque cas d'utilisation.

Ce qu'il reste à faire...

Il reste à gérer les traitements concernant les modules, les sessions et les inscriptions.

BiblioICGO

Programmation objet : 3 classes sont déjà définies stage, stageEtale et stageGroupe (héritage).

Création des classes Module, Session et Inscription à partir du diagramme de classes.

Classe Module

Données privées :

- int numModule
- toutes les autres données de type string (voir la structure de la table dans la BD)
- un module peut être intégré dans plusieurs stages donc création d'une liste de stage nommée List<Stage> lesStages

Constructeurs :

- faire un **constructeur par défaut** avec comme seule instruction l'instanciation de la liste lesStages
- faire un **constructeur** avec les paramètres concernant les données privées **sauf la liste**
- faire un **constructeur** avec les paramètres concernant **toutes** les données privées

Accesseurs :

- faire les **accesseurs get et set** de **toutes** les données privées :
GetNumModule(), SetNomModule(string value), GetNomModule(), SetNomModule(string value), GetNomSupportCours(), SetNomSupportCours(string value), GetNomPresentation(), SetNomPresentation(string value), GetPlaceSupportCours(), SetPlaceSupportCours(string value), GetPlacePresentation(), SetPlacePresentation(string value), GetLesStages(), SetLesStages(List<Stage> value)

Classe Session

Données privées :

- int numSession
- Datetime dateSession
- une session est liée à un seul stage (identification relative dans la BD) donc création d'une donnée de type Stage nommée leStage
- une session est animée par un formateur donc création d'une donnée de type Formateur nommée leFormateur
- une session a lieu dans une agence donc création d'une donnée de type Agence nommée lAgence
- une session accueille plusieurs stagiaires donc création d'une liste de stagiaire nommée List<Stagiaire> lesStagiaires

Constructeurs :

- faire un **constructeur par défaut** avec la seule instruction l'instanciation de la liste lesStagiaires
- faire un **constructeur** avec les paramètres concernant les données privées **sauf la liste**
- faire un **constructeur** avec les paramètres concernant **toutes** les données privées

Accesseurs :

- faire les accesseurs get et set de toutes les données privées :
GetNumSession(), SetNumSession(int value), GetDateSession(), SetDateSession(DateTime value), GetLeStage(), SetLeStage(Stage value), GetLAgence(), SetLAgence(Agence value), GetLeFormateur(), SetLeFormateur(Formateur value), GetLesStagiaires(), SetLesStagiaires(List<Stagiaire> value)

Classe Inscription

Une inscription consiste à inscrire un stagiaire à une session de stage. L'état de l'inscription est défini.

Données privées :

- Session laSession
- Stagiaire leStagiaire
- string etatInscription

Constructeurs :

- faire un **constructeur par défaut**
- faire un **constructeur** avec les paramètres concernant **toutes** les données privées

Accesseurs :

- faire les accesseurs **get** et **set** de toutes les données privées :
GetLaSession(), SetLaSession(Session value), GetLeStagiaire(), SetLeStagiaire(Stagiaire value),
GetEtatInscription(), SetEtatInscription(string value)

BiblioICGODAO

Création et/ou modification des classes ModuleDAO, StageDAO, SessionDAO et InscriptionDAO.

Classe InscriptionDAO

Il est nécessaire de vérifier si l'inscription d'un stagiaire est possible.

Il est souhaitable de créer une procédure stockée **verifierPlaces** dans la base MySQL ayant les paramètres suivants :

- codeCompetence, noStage et noSession en entrée
- dispo : int en sortie
- traitement de la procédure : compter le nombre de stagiaires déjà inscrits à la session
- comparer le nombre d'inscrits avec le nombre de places
- si nbInscrits < nbPlaces alors dispo = 1 sinon dispo = 0

Création d'une méthode **verifierPlacesDisponibles** sous forme d'une fonction qui renvoie un booléen dans la classe InscriptionDAO.

Appeler la procédure stockée dans la méthode verifierPlacesDisponibles et tester la valeur du paramètre de sortie

- si dispo = 1 alors retourner vrai sinon retourner faux

BiblioSupport

La classe Utilitaires contient des fonctions permettant d'extraire les identifiants à partir d'un libellé choisi dans un comboBox d'une IHM. Rien à faire normalement.

BiblioManager

La classe Manager contient des fonctions permettant de charger les informations de la base dans les différents comboBox des IHM.

Programmation des IHM suivantes selon les cas d'utilisation du cahier des charges.

frmModule

ComboBox : affichage numéro + nom

fonction extraireNumModule (libelleModule) : entier

pour modifier ou supprimer un module dans la base (voir code frmStagiaire et classe Utilitaires).

Ajout, modification et suppression des informations dans la base.

Problème avec les données contenant une ' : exemple : l'er...

Il faut placer 2 cotes l"er... afin qu'une seule soit prise en compte.

frmStage

Coder le bouton Affecter Module pour gérer l'appel du formulaire frmAffecterModule (voir code frmFormateur gérant l'appel de frmAffecterCompetence).

frmAffecterModule

liste lstModule : permet de choisir plusieurs modules ; doit contenir tous les modules non affectés au stage choisi dans la comboBox de frmStage.

Le datagridview a été construit par l'intermédiaire des propriétés AllowUserToAddRows et Columns.

Bouton Ajouter : affiche les modules choisis dans le datagridview et supprime les modules dans la liste lstModule.

Ajoute les données dans la base (table COMPOSER) et met à jour la liste et le datagridview (voir code frmAffecterCompetence).

Suppression de modules affectés à un stage :

L'utilisateur peut cliquer sur l'icône de suppression (X) pour un module à supprimer (événement CellContentClick du datagridview dgvModule à programmer).

L'utilisateur peut sélectionner un ou plusieurs modules à supprimer et valide avec la touche SUPPR (événement UserDeletingRow du datagridview dgvModule à programmer).

Supprime les données dans la base (table COMPOSER) et met à jour la liste et le datagridview (voir code frmAffecterCompetence).

frmSessionStage

cboSession = codeCompetence + NumStage + numSession + Intitule

cboStage = codeCompetence + NumStage + Intitule

Possibilité de choisir une session afin d'afficher une session ou un stage pour ajouter une nouvelle session.

cboAgence : contient les agences (nomAgence). Affiche l'agence de la session si choix de session.

cboFormateur : contient les formateurs (numFormateur + nom + prenom) ou les formateurs compétents pour assurer un stage. Affiche le formateur de la session si choix de session.

Nécessité d'utiliser ou d'écrire des fonctions ou des procédures pour :

- extraire l'identifiant d'une session : procédure avec paramètre d'entrée : libellé choisi dans cboSession

- paramètres de sortie : codeCompetence, NumStage, numSession
- extraire l'identifiant d'un stage : procédure avec
paramètre d'entrée : libellé choisi dans cboStage
paramètres de sortie : codeCompetence, NumStage
- extraire l'identifiant d'un formateur : fonction retournant le numéro

Valorisation des zones et comboBox lors d'une session choisie

Nécessité de déterminer l'index de cboStage et cboFormateur pour afficher la valeur adéquate.

Exemple :

comboBox cboFormateur avec un formateur ayant l'identifiant 2

2. Dupont correspond à l'index 0 de la comboBox

3. Martin correspond à l'index 1 de la comboBox

Il est possible d'utiliser l'une des méthodes suivantes :

- int FindString(string chaîne) renvoie la valeur de l'index correspondant au premier élément qui commence par la chaîne
- int FindStringExact(string chaîne) renvoie la valeur de l'index correspondant au premier élément qui représente exactement la chaîne

La valeur de l'index doit être affectée à la propriété SelectedIndex de la comboBox.

frmInscription

cboStagiaire = numStagiaire + nom + prenom

cboSession = codeCompetence + NumStage + numSession + Intitule

Extraction des identifiants lors du choix d'un stagiaire et d'une session.

Les sessions affichées sont celles auxquelles le stagiaire n'est pas encore inscrit.

frmConfirmerInscription

Même comboBox que frmInscription.

Les sessions affichées sont celles auxquelles le stagiaire est inscrit.

Mettre l'état de l'inscription à définitif.