DOCUMENTATION DIAGRAMME DE CLASSE

Rappel : Les attributs privés avec une première lettre en majuscule signifient qu'il y a un attribut privé et une propriété publique, mais nous n'arrivons pas à mettre le +/.
Si l'attribut est publique et a une première lettre en majuscule, cela signifie qu'il n'y a qu'une propriété et pas d'attribut. Elles ont donc toutes un get et set

- La classe Nommable est une classe abstraite qui permet de ne pas avoir à insérer un nom dans chaque classe qui peut en avoir un. Elle contient donc une propriété Nom.

- La classe Utilisateur est une classe fille de Nommable. Un Utilisateur a donc un nom, un prénom, une date de naissance qui ne sera pas affcihée mais un âge ,qui sera affiché,calculé grâce à la la méthode CalculAge qui reçoit sa date de naissance en paramètre. Un Utilisateur a aussi une taille et poids qu'il devra mettre à jour lui même sur l'application, ainsi qu'un identifiant, qui lui est unique et donc qui est la seule variable prise en compte dans les méthodes Equals et CompareTo, et un mot de passe. De plus , il a un Programme dernierprogramme qui est le dernier programme qu'il a effectué, ainsi qu'une Difficulté diffDernierprog qui est la difficulté de ce dernier. Son constructeur prend en paramètre un nom, prénom, une date de naissance, une taille, un poids, un identifiant et un mot de passe. Finalement, la méthode LancementProgramme met à jour le dernierProgramme choisi ainsi que sa dernière difficulté.

- La classe Administrateur est une classe fille de la classe Utilisateur. Elle n'apporte rien de plus niveau code, mais elle permettra à la connexion d'un utilisateur d'ouvrir la fenêtre normal si l'utilisateur qui se connecte est seulement un utilisateur, ou la fenêtre principale d'administrateur s'il est admin.

- La classe Programme est une classe fille de Nommable. Elle a de plus, comme attributs, une description, un chemin d'image et une LinkedList d'Exercice lesExercices. La propriété calculable nbExercices n'est pas forcément la plus utile puisqu'il s'agit uniquement au premier regard de la longueur de la liste d'exercice. Mais à la création d'un programme depuis la fenêtre, un nombre d'exercice est à rentrer pour que la création du programme fonctionne. Ce nombre d'exercice est donc d'abord un nombre rentré par l'administrateur qui veut ajouter un programme à ceux déjà présent, et ensuite une propriété calculable si l'admin supprime un exercice à un programme ou en ajoute un. Son constructeur reçoit en pramètre uniquement un nom, une description et un chemin d'image, et instancie la LinkedList lesExercices. Elle ne possède que deux méthodes AjouterExercice et supprimé Exercice qui reçoive toutes deux un exercice en paramètres.

- La classe Exercice est une classe fille de Nommable. Elle a de plus un chemin d'image et 4 valeurs : valeurDeb pour la difficulté DEBUTANT, valeurInter pour la difficulté INTERMEDIAIRE, valeurExpert pour la difficulté EXPERT et valeurCourante qui est la Valeur prise pour l'exercice en fonction de la difficulté qu'un utilisateur a choisi au lancement d'un programme.

- La classe Valeur contient 3 entiers : un nombre de séries, un nombre de répétitions et un temps de repos qui varieront en fonction de la difficulté.

- La classe Listes est la classe qui permettra la sérialisation. Elle est divisé dans le code en deux parties ce qu'il fait qu'elle est partielle. La classe Listes tout court contient un Dictionnaire ,prenant en clé un identifiant et en valeur un utilisateur ,listCompte. Elle contient aussi la LinkedList de tous les programmes listProgramme. On peut l'instancier en passant en paramètre soitt le dictionnaire, soit la linkedlist, soit les deux soit aucun. Les noms de méthodes sont assez explicites : RecherUtilisateur renvoie un utilisateur en fonction d'un login passé en paramètre, si le login est introuvable, elle renvoie null. VérifierConnexion renvoie false si le login ne correspond à aucun utilisateur ou si le login de correspond pas au mot de passe, sinon true. AjouterUtilisateurInscirption ajoute un utilisateur au dictionnaire de compte, AjouterProgramme ajoute un programme passé en paramètre à la LinkedList listProgramme et SupprimerProgramme en supprime un. AjouterUnExercice ajoute un exercice à la liste des programmes.
  La classe Listes.DonneesCourantes contient elle l'utilisateur connecté ainsi que le programme choisi pour faciliter le DataBinding. Sa méthode LancementProgramme met à jour la valeurCourante des exercices contenus dans la liste des exercices du programme passé en paramètre en fonction de la difficulté passé en paramètre et appelera la méthode LancerProgramme d'un Utilisateur.

- La classe Manager ne contient que des méthodes static qui reçoivent toutes une Listes en paramètre en plus d'autres paramètres. Elles sont les méthodes qui vérifient et valide les objets de Classe que nous avons créés et appelle les méthodes de la classe Listes qui portent le même nom qu'elles.

- La classe CreationObjectValidator ne contient que des méthodes static qui renvoient un bool. Elles vérifient si les attributs des objets passé en paramètres sont correctes (pas vides pour les string ou pas trop élevé/bas pour les nombres).

- La classe abstraite DataLoad contient un chemin d'où les données devront être chargées. Sa méthode abstraite ChargeDonnes renvoie une Listes et devra être modifié en fonction du support avec lequel on charge les données (fichier binaire, base de données..)

- La classe abstraite DataSave contient un chemin qui est l'endroit où les données devront être sauvegardées. Sa méthode abstraite SauvegardeDonnes prend une Listes en paramètres et devra être modifié en fonction du support sur lequel on sauvegarde les données (fichier binaire, base de données..)

- La classe StubData est une classe fille de DataLoad. Il contient des méthodes qui charge un dictionnaire d'utilisateur,3 liste d'exercice qui seront les LinkedList lesExercices de 3 programme qui seront eux mêmes mis dans une Liste. Sa méthode ChargeDonnes appelle ces différentes pour retourner une liste.