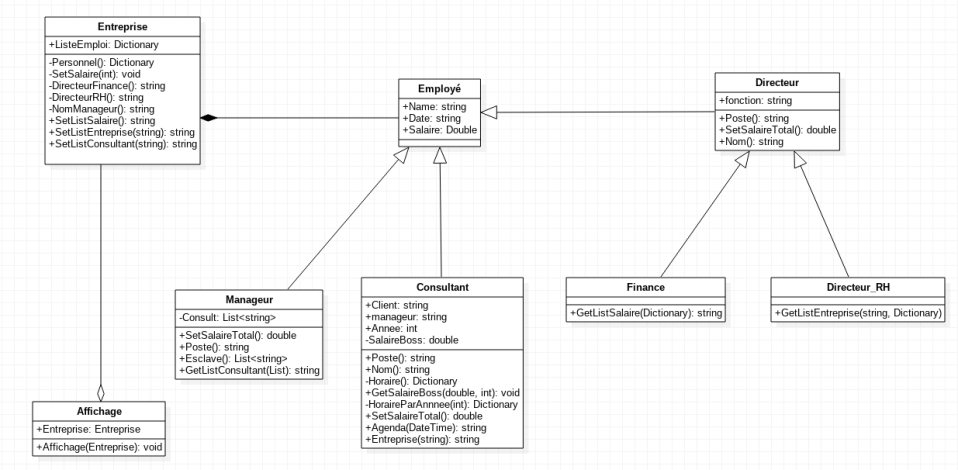
Travail 1: Système de gestion des employés d'une entreprise

Julien Stilmant 14054 Thierry Frycia 14212

27 octobre 2017

1 Structure

1.1 Diagramme



1.2 Fonctionnement du code

1.2.1 Structure générale

La fonction `Main()` qui se trouve dans la classe `MainClass` dans le fichier `Execution.cs` appelle la fonciton `Emploi()` qui est définie juste en dessous. Cette fonction va lire le contenu des fichiers json, l'interprète et mémorise les données dans les objets du type aproprié.

Après ça, la fonction `Affichage()` est appellée avec en paramètre le dictionnaire qui contient tous les objets déserialisés auparavant. C'est cette fonction qui fait l'interface avec l'utilisateur et qui va par après appeler les méthodes appropriés sur les objets pour obtenir leurs attributs.

1.2.2 Classes principales

Consultant La classe `Consultant` est une sous-classe de `Employé`. Elle hérite donc simplement des attributs `Name`, `Date` et `Salaire`. En plus de ça, elle génère des attributs personnels `Client` et `manageur`.

Le calcul du salaire se fait dans la méthode `SetSalaireTotal`, qui vérifie si les missions ont été faites dans une boîte externe ou non et prends en compte le salaire du manager.

Directeurs `Directeur_RH` et `Finance` héritent tous les deux de la classe `Directeur` qui définit les attributs pour la fonction et le salaire de la personne.

La classe `Directeur_RH` contient une méthode `GetListEntreprise()` qui renvoie les noms et les dates des consultants qui ont travaillé dans une certaine entreprise. Cette méthode sera appelé par la méthode `ListConsultants()` dans la classe `Entreprise`.

Le même principe est utilisé pour la méthode `GetListSalaire()` du directeur des finances.

Manageur Les deux méthodes principales de cette classe sont `SetSalaireTotal()` et `GetListConsultant()`.

La première calcule le salaire en fonction du nombre de consultants sous sa responsabilité.

La deuxième est utilisée par la méthode `SetListConsultants()` dans la classe `Entreprise` pour générer le rapport du manageur de la même façon que pour les directeurs.

Entreprise `SetSalaire()` envoie le salaire des manageurs à leurs consultants pour que ceux-ci puissent calculer leur salaire.

`DirecteurFinance()`, `DirecteurRH()` et `NomManageur()` renvoient le nom de la personne pour pouvoir l'afficher au final.

`SelListSalaire()` SelListSalaire envoie à la classe `Finance` un dictionnaire des noms et des salaires pour que l'intance de cette classe puisse renvoier la liste des salaires.

`SelListEntreprise` envoie à la classe `Directeur_RH` un dictionnaire des noms et des dates pour que l'intance de cette classe puisse renvoier la liste des consultants travaillant dans une entreprise donnée.

`SelListConsultant` envoie à la classe `Manageur` une liste des noms des consultants et de lentreprise où ils travaillent pour que l'intance de cette classe puisse renvoyer cette liste.

2 Fichiers

Toutes les données sont stockées dans des fichiers texte au format `json` séparés selon le poste ou la fonction.

Dataadress.json Contient la liste des emplacements des fichiers `json` correspondant aux employés.

Dataemployé.json Contient une liste de dictionnaires chacun reprenant le nom, le poste, la date de naissance, le salaire et une liste des personnes associés.

Data(nom).json Chaque fichier représente un consultant et contient la liste de ses missions avec leur date de début et de fin.

3 Interface

Au lancement, l'interface vous propose les 3 choix d'actions possible, c'est à dire les 3 types de rapports possibles à générer.

Par exemple, si vous voulez générer un rapport pour le directeur des ressources humaines, choisissez la deuxième option en écrivant `2` dans la console. On vous demandera ensuite pour quelle société vous voulez lister les consultants. Entrez par exemple `Google` ou `Thales` pour générer le rapport.