Rendu de TP

HAI913I - TP1 Partie 2

BOURRET Maxime

1 Travail réalisé

1.1 Exercice 1

Pour le premier exercice, j'ai donc récupéré le code de la partie 1 du TP pour avoir une base. J'ai transformé la classe `Parser` en enlevant le main et en ajoutant des constructeurs, setters et getters afin de pouvoir instancier un parseur. J'ai ensuite modifié les visiteurs pour obtenir un visiteur par type nécessaire à savoir : package, classe, déclaration de méthode, invocation de méthode. Ensuite j'ai créé un main qui réutilise le code du main de l'ancienne classe parseur. On demande d'abord de chemin projet à analyser, puis après avoir ouvert chaque fichier du dossier *src*, construit de chacun et on accepte chaque visiteur. D'autres traitements comme le comptage du nombre de lignes sont effectués puis on affiche les résultats question par question en utilisant les méthodes des visiteurs et de la classe main.

1.2 Exercice 2

Pour le second exercice, j'ai créé un classe `CallGraph` qui contient un seul attribut *callGraph* de type `Map<String, List<String»`. Puis en réutilisant le code de la partie 1, pour chaque méthode visitée par le visiteur de déclaration de méthode, on appelle le visiteur d'invocation de classe. Ensuite, on va récupérer les métthodes appelées par la fonction et les insérer dans la `Map` du graphe d'appel avec comme clé le nom de la méthode appelée et comme valeur le nom de la méthode appelée. Une fois toutes les méthodes parcourues, on va afficher le graphe dans la console sous forme d'objet JSON en affichant le nom de la classe déclarée suivie d'une array avec tous les noms des méthodes invoquées par cette dernière.

Comme remarque personnelle sur cette partie du TP, cela m'a rappelé mon stage de cet été lorsque dans l'équipe de recherche de Houari SAHRAOUI, j'ai été amené à extraire un graphe d'appel d'un programme dummy et pour lequel ma collaboratrice et moi avons utilisé `JProfiler` puis avions appliqué des traitements de texte avec Python pour créer l'objet JSON.