SYNTHÈSE

ENSG Géomatique

- Organisation / Répartition
- Explications du code
- Cheminement
- Conclusion

Wandrille FALALA Typhaine JOUAN

ORGANISATION RÉPARTITION

✓ Points Positifs :

- ➡ Tenue d'un Cahier de Bord
  - Utile dans la phase d'analyse pour lister et garder en mémoire toutes les hypothèses
  - Utile également pour savoir ce qu'il reste à faire, ce qui n'a pas été fini
  - Aide à signaler et à garder une trace des modifications à effectuer
- ➡ Utilisation de Google Drive
  - Pas de risque de perte de nos données comme sur une clé USB ou un disque dur
  - Pas besoin de se familiariser à un nouvel environnement type GitHub pour le moment
  - Garde en archive les anciennes versions du code
- ➡ Briefing de début de séance
  - Nous a aidé à répartir de façon équitable et réaliste le travail en début de séance, et adapter les prédictions de notre diagramme de GANTT à la réalité de la programmation

◉ Contraintes :

- ➡ Travail haché
  - Les vacances de Pâques et la semaine de Mai nous on coupé dans notre élan
  - Il était plus compliqué pour nous de coder et nous organiser à distance
- ➡ Obligation de modifier le nom des fonctions et de la base de données parfois par rapport à la phase analyse pour des raisons de simplification
  - Mène à modifier beaucoup de fonctions quand on modifie un attribut

EXPLICATIONS DU CODE

Code typographie : **package** / classe

Collector est composé de trois parties, représentées dans l'implémentation Java par trois packages.

Le premier, et le plus vital, est **coeur_collector**, il contient toutes les classes qui sont partie intégrante du projet, on y trouve :

- Utilisateur qui est la classe mère de Administrateur et de Eboueur
- Collecte qui stocke toutes les informations relatives aux collectes, et qui est géré par l'Administrateur
- Connexion qui sert à se connecter à la Base de Données essentiellement
- Equipe et Vehicule qui sont des classes indispensables pour réaliser les collectes

Le second est **Carte** qui permet de réaliser une des attentes primordiales du projet, à savoir afficher la carte des collectes, avec les secteurs (représentant les quartiers) et les trajets. Cette carte est visualisable par tous les Utilisateurs (Administrateurs et Eboueurs). Elle comporte différentes classes :

- Carte qui permet de créer la fenêtre de la classe puis de dessiner la carte grâce à java.swing.JFrame (Java2D)
- FormeGeometrique permet de faire les dessins sur la carte et est la classe mère de Decharge, Secteur et Trajet. Ces trois classes héritent de FormeGeometrique car les décharges sont des points avec un identifiant et des coordonnées
- Dessin et Dessin_equipe sont les outils qui permettent à Carte d'afficher la carte

Le dernier est **Statistiques** dans lequel on trouve toutes les classes relatives aux statistiques que l'Administrateur peut voir. Les classes qui le composent sont :

- Stat_collecte qui fait des statistiques inhérentes aux collectes, surtout des moyennes sur le poids récolté
- Stat_eboueur qui fait regroupe le temps de travail travaillé dans le mois et le salaire
- Stat_equipe qui fournit des statistiques mensuelles sur chaque équipe, donc les entrées dans la table de données associée sont les statistiques par équipe/par mois
- Stat_secteur qui fournit des statistiques annuelles liées au secteur

Enfin, le package supplémentaire **Main** est celui qui permet d'utiliser l'application Collector, il est composé d'une unique classe Main qui fait appel successivement à toutes les méthodes des différents packages. La première interaction est l'authentification, soit en tant que Administrateur, soit en tant qu'Eboueur. Si c'est un Eboueur, il ne peut que visualiser la carte des collectes. Si c'est un Administrateur, il aura alors accès à toutes les fonctionnalités de modification de la base de données, ainsi qu'aux différentes statistiques et à la carte.

CHEMINEMENT / DIFFICULTÉES

CHEMINEMENT

- Nos premières séances ont été dédiées uniquement à la partie analyse, nous avons souhaité discuter des limites de notre sujet et notre organisation de code avant de nous lancer.
- Pour coder, nous avons commencé par les classes du package **coeur_collector**. En effet, elles nous semblaient primordiales.
- Puis, nous nous sommes réparti les tâches, l'un codait la partie **Carte** pendant que l'autre s'occupait des **Statistiques**.
- Dans le package **Carte** il y avait également la partie Java2D, nous avons décidé de nous renseigner et d'apprendre chacun de notre côté avec des tutoriels internet pour ensuite le coder.
- Finalement, une fois toutes les méthodes et classes implémentées, il restait à faire le **Main** pour faire la liaison entre tout le reste.
- Enfin, nous avons exporté notre projet dans Eclipse en « Runnable JAR File Export » pour avoir une console accessible même sans Eclipse.

DIFFICULTÉES

- Pendant notre code, nous avons rencontré des soucis de gestion des dates avec SQLite3 car pas de type date comme sous PostGre
  - Il a donc été nécessaire de faire des fonctions annexes, pour duree par exemple car la comparaison de dates dans Java n'était pas gérable.
- Mais aussi des soucis avec les changements de statuts (disponibilité) des différents objets tels que Eboueur, Equipe, Vehicule. C'est pour cela que l'on ne peut savoir dans notre code que la disponibilité d'un objet au moment de la requête.
  - On aurait pu faire une classe Calendrier où stocker des booléens dans chaque case du Calendrier pour chaque objet en fonction de sa disponibilité

CONCLUSION

Nous sommes satisfaits de notre code source et de l'organisation de notre travail, mais le fait de n'avoir pu travailler que lors des séances à l'école et pas pendant les vacances nous a empêché de développer une vraie interface pour l'application.

Il nous a été aussi très satisfaisant de faire un projet de bout en bout, de la modélisation à la réalisation. Cela a été bénéfique d'une part pour notre esprit de synthèse pour les diagrammes UML mais aussi pour notre niveau en programmation Java.