`Editor: Yuxin SHI` `SI 2 FISE`

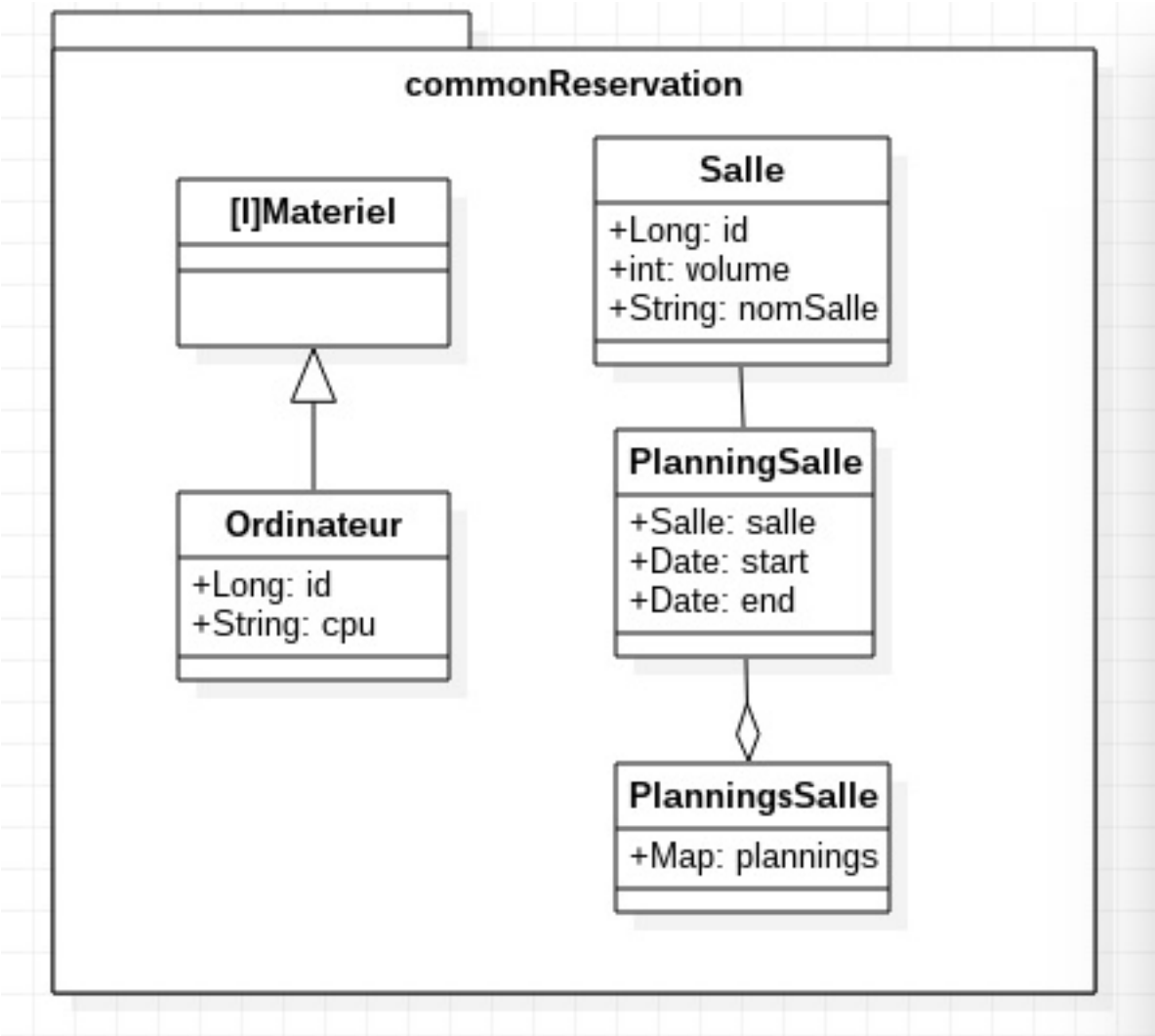
Compte rendue : Outils pour cycle de vie Logiciel

TP4 & TP5

Ajouter Planning

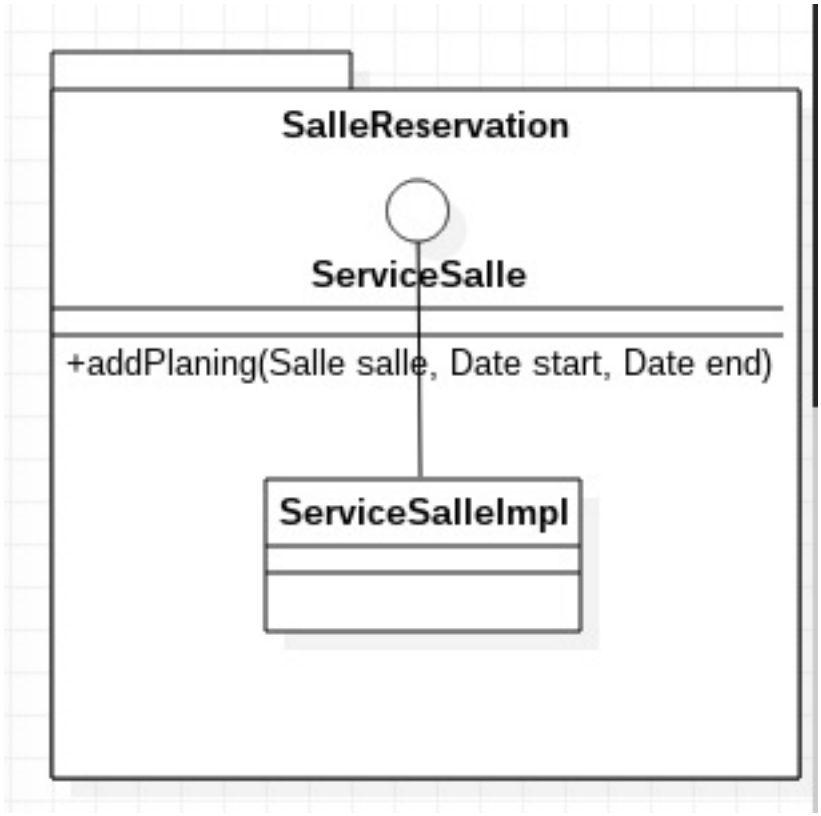
L'adresse de Git: https://github.com/yuxinvalo/TPGit_Outil

Dans le commonReservation, il y a les entités du Matériel et de la Salle, j'ai ajouté aussi deux class: Planning et Plannings.
Ceci un schémas de module commonReservation:



J'ai fait juste le planning pour la salle, PlanningSalle contient juste un planning pour une salle, et PlanningsSalle contient une collections de PlanningSalle, j'utilise Map pour faciliter la recherche, et j'utilise "Standalone" pour rassurer qu'il y a qu'une plannings pour les salles dans le processus ainsi faciliter d'ajouter les plannings sans conflit du temps pour chaque salle.

Ceci un schémas de module reservationSalle:



Il y a qu'une service selon tp, addPlaning retourne un boolean si on a réussi à ajouter un planing pour une salle.

Pour le module serveurReservation, semble qu'on doit utiliser le mode Factory pour générer les réponses des requête du client, mais je ne vois pas comment faire ça. Ce module dépend de commonReservation et reservationSalle, dans la classe ReservationFactory, ServiceSalle devrait être instancier, mais je comprend pas trop. Ou ça veut dire que c'est la partie du client?

A mon avis, c'est la classe ServiceReservation qui va instancier une service addPlaning, et le ReservationFactory fabrique juste le code de HTTP si le service a bien effectué.
Ceci le code de ReservationService:

```
@RequestMapping(method=RequestMethod.GET)

public @ResponseBody ReservationFactory addReservation(@RequestBody Planning planning
   ServiceSalle serviceSalle = new ServiceSalleImpl();
   boolean bool = serviceSalle.addPlanning(planning);
   return new ReservationFactory(bool);

   }
```

Ceci le code de ReservationFactory:

```
public class ReservationFactory {

    public ReservationFactory(boolean bool){
        if(bool){
            success();
        } else {
            fail();
        }
    }

    public ResponseEntity<HttpStatus> success(){
        return ResponseEntity.ok(HttpStatus.OK);
    }

    public ResponseEntity<HttpStatus> fail(){
        return ResponseEntity.ok(HttpStatus.CONFLICT);
    }
}
```

Git

Usage basique

Ce projet je le fais toute seule, donc je ne peux que faire une branche et en merger avec la branche principale.

```
# create a new branch
git checkout -b yuxin

# merge with master
git merge master

# push to remote repository
git push remote yuxin
```

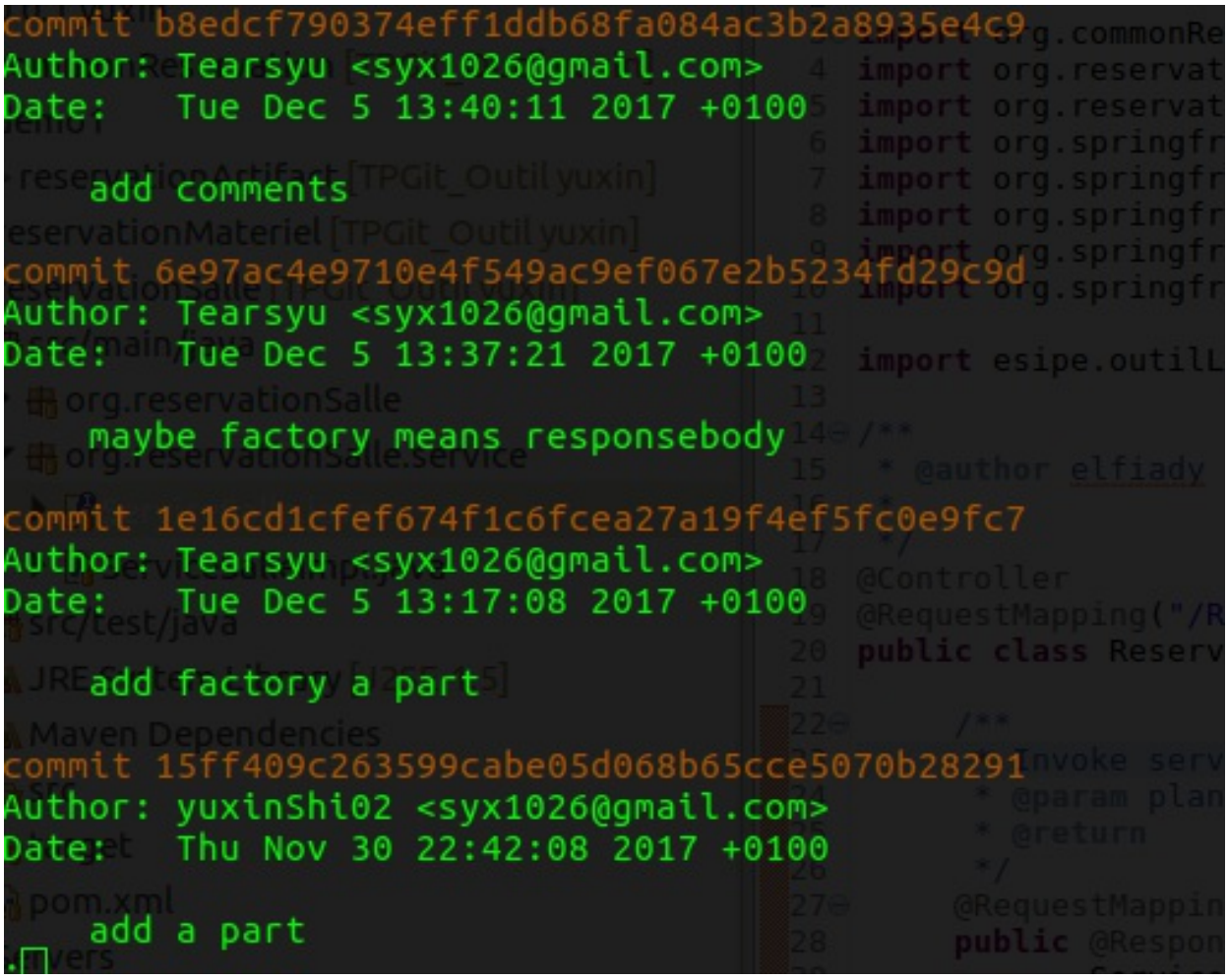
Conflit

Manuel

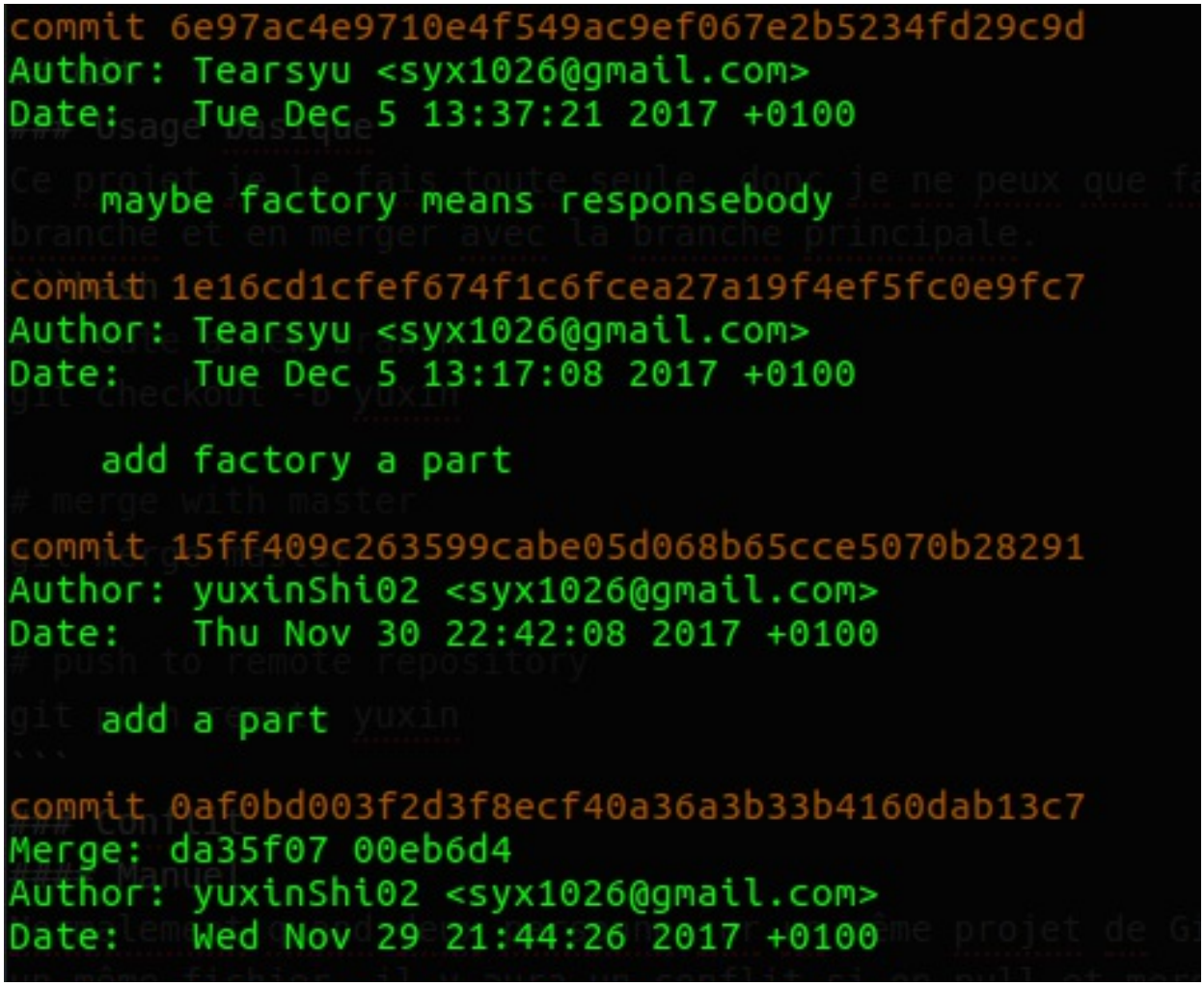
Normalement quand deux personne sur un même projet de Git modifient un même fichier, il y aura un conflit si on pull et merger, si on veut garder ma version, il faut supprimer l'index de conflit manuellement dans ce fichier.

Reset et Revert

Ceci mon git log:



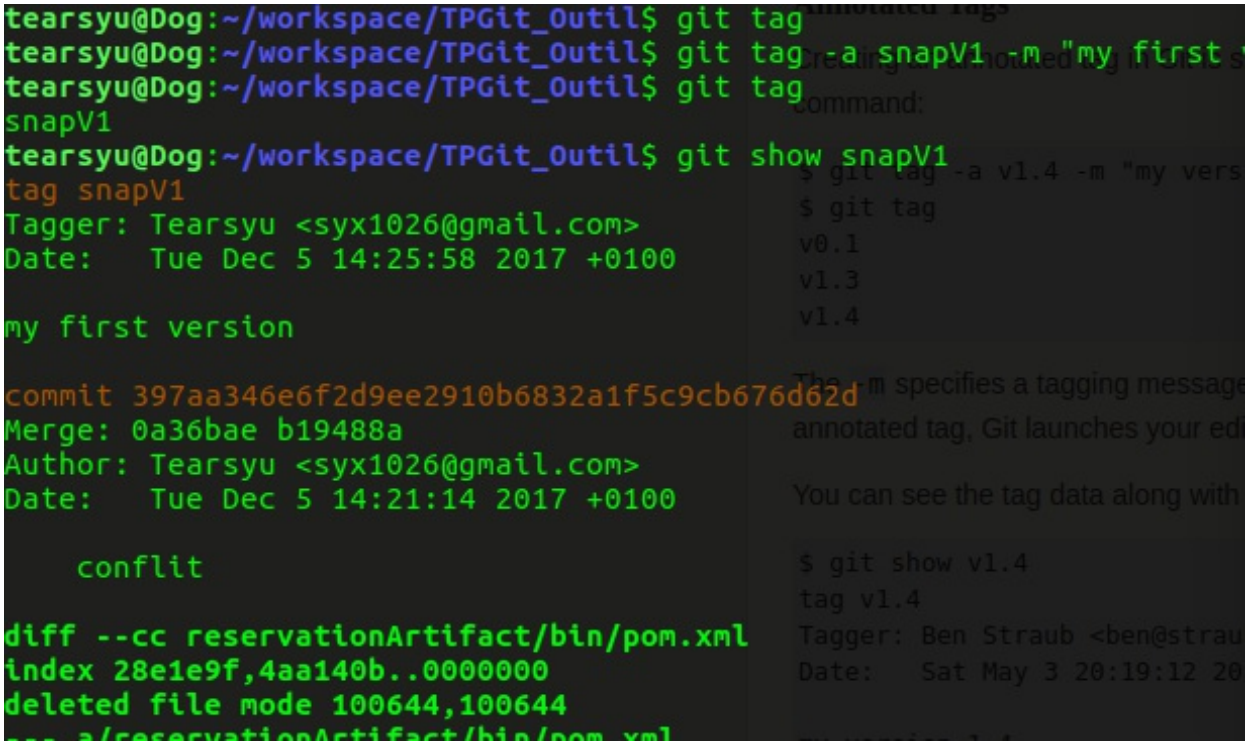
Je vourdrais retourner à commit "6e97ac" et mettre le dernier commit dans la poubelle on peut utiliser mais sans changer le fichier actuelle: `git reset --soft HEAD^`, maintenant le commit "b8edcf" n'existe plus:



Si on ne veut pas perdre le dernier commit en retournant HEAD^, on peut aussi utiliser `git revert HEAD^`, git va créer un nouveau commit dans cette branche qui correspondant avec HEAD^.

Tag

Ajouter un tag pour cette version:



Rebase

Rebase fonctionne un peu comme merge, mais il va ajouter les commits souhaités de la branche que l'on veut merger dans la branche curante. On peut l'utiliser pour notre branche personnelle pour aligner et nettoyer l'histoire de git log, mais ce n'est pas une bonne idée de l'utiliser dans une branche publique, ça peut être risqué parce que on est obligé de merger encore une fois pour notre master et celui des autres du groupe.