TD Pratiques XP 2

Coffre à Trésor (XP en mini-projet) https://classroom.github.com/a/xhSVWf3J

Organisation de la séance

Durée totale : 4 heures. Travail en **équipes fixes de 4/3 personnes**.

Chaque équipe fonctionne comme une petite équipe XP :

- deux binômes travaillent en parallèle ;
- rotation des binômes toutes les 25 à 30 minutes ;
- à chaque rotation : échanges, revue rapide, synchronisation.

Le but est que l'équipe de 4 vive une dynamique de mini-projet XP :

- communication constante,
- vision partagée du backlog,
- cohérence du design,
- alignement sur les tests.

Fonctionnement d’une équipe de 4

Constitution

Une équipe = **4 personnes**. Deux binômes actifs à tout moment.

Rôle du binôme

Travail classique de Pair Programming :

- Driver : écrit le code.
- Navigator : pense aux tests, au métier, au design.

Rotation Driver/Navigator régulière.

Rôle de l'équipe

Toutes les 25–30 minutes :

- les 4 se regroupent,
- mini-synchronisation (5 minutes max),
- rotation des binômes (on mélange les paires),
- alignement sur :
    1. les tests écrits,
    2. les comportements couverts,
    3. la conception en cours,
    4. les prochaines micro-fonctionnalités.

L'équipe doit maintenir une cohérence globale malgré les rotations.

Consignes spécifiques pour les équipes de 4

1. Vision partagée

Chaque équipe doit :

- maintenir une liste des comportements déjà testés,
- maintenir une compréhension commune du backlog,
- garder un code simple compréhensible par les 4 membres.

2. Synchronisation régulière

La réunion de synchronisation (toutes les ~30 minutes) sert à :

- discuter du design,
- éviter les divergences,
- repérer les doublons de tests,
- se réaligner avant de reformer de nouveaux binômes.

3. Aide interne avant aide externe

En cas de blocage :

```
1. le binôme demande d'abord aux deux autres membres de l'équipe ;
2. si l'équipe entière ne débloque pas → intervention de l'enseignant ;
*l'enseignant peut fournir un test précis pour relancer.*
```

4. Propreté collective

Chaque membre est responsable :

- de la lisibilité du code,
- de la cohérence des tests,
- de supprimer la duplication,
- de refactoriser quand nécessaire.

Travaillez « comme si quelqu'un d'autre reprenait votre code après la pause » (car c'est le cas).

Rappels XP adaptés aux équipes de 4

Pair Programming

- binômes courts, rotation rapide ;
- communication maximale ;
- design volontairement simple ;
- pas d'anticipation : YAGNI.

Test-Driven Development

- un petit test à la fois ;
- un comportement à la fois ;
- pas d’écriture de code sans test en échec ;
- refactorisation collective.

Collective Ownership

Le code appartient à l'équipe entière :

- chacun peut modifier n'importe quelle partie,
- l'équipe doit maintenir une cohérence de style et de structure.

Sujet du TD : Le Coffre à Trésor

Vous devez implémenter un module représentant un coffre à trésor. Le coffre contient des objets (nom, poids, valeur) et possède une capacité maximale.

Vous devez avancer par **micro-fonctions**, guidées par les tests.

Backlog

MVP

US1 : Ajouter un objet

En tant qu’aventurier, je veux ajouter un objet dans mon coffre, afin de stocker du matériel utile.

US2 : Retirer un objet

En tant qu’aventurier, je veux retirer un objet par son nom, afin de récupérer ce dont j’ai besoin.

US3 : Lister les objets

En tant qu’aventurier, je veux connaître la liste des objets présents dans le coffre, afin de voir ce qu’il contient.

US4 : Connaître le poids total

En tant qu’aventurier, je veux connaître le poids total des objets, afin d’évaluer la charge transportée.

US5 : Connaître la valeur totale

En tant qu'aventurier, je veux connaître la valeur totale du contenu, afin d'évaluer la richesse du coffre.

Fonctionnalités utiles

US6 : Empêcher les doublons

En tant qu'aventurier prudent, je veux qu'il soit impossible d'ajouter deux objets portant le même nom, afin d'éviter les incohérences.

US7 : Fixer une capacité maximale

En tant qu'aventurier, je veux configurer la capacité maximale du coffre, afin d'éviter qu'il ne soit trop lourd.

US8 : Gérer les objets rares

En tant qu'aventurier, je veux que les objets rares valent le double, afin de valoriser leur rareté.

US9 : Trier les objets

En tant qu'aventurier, je veux trier les objets par nom, poids ou valeur, afin d'y voir plus clair.

US10 : Rechercher un objet

En tant qu'aventurier, je veux trouver un objet par son nom, afin de le localiser rapidement.

Bonus

US11 : Verrouiller le coffre

En tant qu'aventurier, je veux pouvoir verrouiller mon coffre, afin d'empêcher toute modification.

US12 : Historiser les actions

En tant qu'aventurier, je veux que chaque action (ajout, retrait, échec) soit historisée, afin de pouvoir retracer les événements.

US13 : Transférer un objet

En tant qu'aventurier, je veux transférer un objet d'un coffre A à un coffre B, afin de gérer plusieurs coffres facilement.

Bilan

Peut inclure :

- cohérence du code entre les différents binômes,
- pertinence et couverture des tests,
- qualité des synchronisations toutes les 30 minutes,
- capacité à travailler sur du code écrit par d'autres membres,
- respect du Pair Programming et du TDD.