TD Pratiques XP

Développement d’un “Pendu minimal” en XP

Objectif de la séance

Vivre une mini-itération XP complète, à deux, en développant une première version (MVP) du jeu du Pendu.

Le but n’est pas de produire un jeu sophistiqué, mais de pratiquer les principales disciplines d’eXtreme Programming :

- User Stories
- Planning Game
- Décomposition par valeur
- TDD : Test-Driven Development
- Pair Programming (Driver / Navigator)
- Design émergent
- Feedback loops courtes
- Rétrospective d’itération

1. Organisation générale de la séance

La séance est structurée comme une vraie itération XP :

- Planning Game
- Sélection du MVP
- Développement en Pair Programming + TDD
- Livraison / Démonstration
- Rétrospective

Vous travaillez en binôme pendant toute la séance.

Les rôles (Driver / Navigator) alternent régulièrement.

2. Rôles en Pair Programming

Driver

- écrit le code au clavier
- suit les instructions du Navigator
- ne prend pas d’initiative non discutée

Navigator

- lit les tests et guide les décisions
- observe la structure générale
- pense aux prochaines étapes
- ne touche pas au clavier

Les rôles changent toutes les 5 à 7 minutes.

3. Le Projet : Pendu – MVP

Vous développerez uniquement le moteur du jeu : pas d'IHM graphique, pas de dessin du pendu.

MVP attendu

Votre version minimale doit permettre de :

- choisir un mot secret (fourni au constructeur ou fixé dans le code)
- proposer une lettre
- savoir si la lettre est correcte ou incorrecte
- compter les erreurs
- détecter :
- victoire (toutes les lettres trouvées)
- défaite (nombre max d'erreurs atteint)

- gérer les doublons :
    - lettre correcte déjà proposée
    - lettre incorrecte déjà proposée

Facultatif, non demandé

- interface console
- choix dynamique du mot
- dessin du pendu
- tout ce qui n'est pas indispensable au MVP

4. Planning Game

1. Écriture des User Stories

- Format attendu : En tant que … je veux … afin de …
- Une fonctionnalité = une story.

2. Estimation rapide

Chaque story est classée en :

- S (simple)
- M (moyen)
- L (complexe)

3. Priorisation

Les stories sont triées par valeur :

- Essentiel

- Utile
- Bonus

TODO : Sélection des 2–3 stories constituant l'itération du jour.

5. Développement (2h15) – TDD + Pair Programming

Travail en cycles très courts :

- écrire un test simple
- le faire échouer
- écrire le minimum de code pour le faire passer
- refactorer
- recommencer

Rappels XP

- Pas de gros tests : avancer par petits pas.
- Le refactoring fait partie du cycle.
- Simplifier si vous êtes bloqués.
- Le code doit rester propre en continu.

Changement Driver/Navigator toutes les 5 à 7 minutes.Toutes les 20 minutes : mini-pause de 2 minutes pour vérifier votre communication et votre progression.

6. Livraison et Démonstration

Chaque binôme montre :

- les tests écrits
- le moteur du pendu qui fonctionne
- comment les tests ont piloté la logique
- les choix de design

Une démonstration XP n'évalue pas la quantité de code, mais :

- la clarté
- la modularité
- la cohérence du TDD
- la communication dans le binôme

7. Rétrospective (20 min)

Questions possibles :

- Qu'est-ce que TDD a changé dans votre manière de coder ?
- À quel moment le Pair Programming a aidé ? Ou gêné ?
- Quelles stories étaient mal découpées ?
- Qu'a révélé la démo sur votre design ?
- Que changeriez-vous pour la prochaine itération ?

L’objectif est d’améliorer la manière de travailler, pas seulement le code.

8. Livrables attendus

À la fin de la séance, vous devez avoir :

- un moteur de jeu minimal qui passe les tests
- une série de tests unitaires écrits en TDD
- un mini-board de stories (photo ou fichier)
- une rétrospective écrite (5–10 lignes)

9. Règles XP à respecter pendant toute la séance

- Tests d’abord, jamais après
- Communication constante dans le binôme
- Petits pas
- Feedback rapide
- Pas de Big Design Upfront
- Pas de code sans test