

# Présentation XP

## Introduction

Les piliers de l'eXtreme Programming (XP) reposent sur des mécanismes clés pour :

- Optimiser l'efficacité des équipes (pair programming, intégration continue, etc.),
- Raccourcir les boucles de feedback (livraisons fréquentes, tests automatisés, rétrospectives),
- Fluidifier les releases (déploiement continu, petites itérations incrémentales),
- Renforcer la collaboration client (user stories, planification adaptative, démonstrations régulières).

La maîtrise de XP passe par une compréhension hiérarchisée :

- Ses Valeurs (communication, simplicité, feedback, courage, respect) → fondations culturelles.
- Ses Principes (ex : "Embrasser le changement", "Livrer de la valeur rapidement") → cadrage stratégique pour choisir/adapter les pratiques.
- Ses Pratiques (TDD, refactoring, CI/CD, velocity tracking, etc.) → outils concrets pour implémenter les principes.

Adaptation contextuelle :

XP n'est pas un framework rigide. Certaines pratiques (ex : pair programming) peuvent être ajustées ou combinées (avec Scrum/Kanban) selon :

- La taille de l'équipe (startup vs. scale-up),
- La criticité du projet (MVP vs. système embarqué),
- La maturité technique (legacy code vs. greenfield).

XP Agile DevOps TDD CleanCode XP XP XP XP

## Valeurs XP

Les **valeurs XP (eXtreme Programming)** définissent la culture et les comportements clés pour des équipes **agiles et techniques**.

### Tableau des Valeurs XP

Valeur	Définition Technique
Communication	<b>Échanges structurés</b> pour :
	- *Knowledge sharing* : <b>mob programming</b> , docs collaboratives (ex : Confluence/Notion).
	- *Résolution de problèmes* : <b>stand-ups techniques</b> , canaux dédiés (Slack/Teams).
	- *Coordination* : <b>planning poker</b> , raffinement de backlog.
Simplicité	<b>Approche minimaliste</b> :
	- *YAGNI* : <b>"You Aren't Gonna Need It"</b> → pas de sur-ingénierie.
	- *DTSTTCPW* : <b>"Do The Simplest Thing That Could Possibly Work"</b> .
	- *Refactoring* : <b>incrémental</b> (ex : *boy scout rule*).
Feedback	<b>Boucles courtes et automatisées</b> :
	- *Tests* : <b>TDD/BDD</b> (feedback en ms).
	- *Demos* : <b>Sprint reviews</b> (validation client en jours).
	- *Rétros* : <b>Amélioration continue</b> des processus.

Valeur	Définition Technique
Courage	<b>Décisions techniques audacieuses :</b>
	- Remettre en question les choix (ex : "Ce framework est-il adapté ?").
	- *Fail fast* : <b>Spikes</b> pour valider des hypothèses avant le dev.
	- *Blameless culture* : <b>Postmortems</b> sans jugement.
Respect	<b>Collaboration bienveillante :</b>
	- *Pair programming* : <b>Montée en compétence</b> (juniors/seniors).
	- *Code reviews* : <b>Focus sur l'amélioration</b> , pas la critique.
	- *Retros actionables* : <b>Écoute active</b> des feedbacks.

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/eadl/bloc3/xp/chap1?rev=1763861954>Last update: **2025/11/23 02:39**