Design patterns essentiels pour chaque projet

Durée: 1 jour
Public cible: Professionnel TI

À qui s'adresse cette formation ?

Concepteurs, conceptrices, architectes et personnes développant des applications

Objectifs de la formation

- Décrire les différentes catégories de patterns
- Appliquer les meilleures pratiques architecturales par des solutions éprouvées dans vos applications
- Identifier les patterns incontournables pour tout projet

Description sommaire

Les design patterns (patrons de conception) sont des solutions orientées objet connues et éprouvées pour des problèmes de conception récurrents. Ils ont l'avantage d'être indépendants du langage de programmation, assurent une modularité du système et permettent une facilité d'évolution. Ils représentent un vocabulaire commun à l'expertise des concepteurs et favorisent le transfert de connaissances entre les concepteurs experts et novices.

Contenu du plan de cours

- Introduction au catalogue de Gamma : Termes et concepts, éléments essentiels et espace des patrons de conception
- Patterns de construction : Fabrication (Factory Method), Fabrique abstraite (Abstract Factory), Monteur (Builder) et Singleton
- Patterns de structuration : Adaptateur (Adapter), Composite, Façade et Procuration (Proxy)
- Patterns de comportement : Commande (Command), Itérateur (Iterator), Observateur (Observer), Stratégie (Strategy) et Patron de méthode (Template Method)
- Technique pratique de sélection d'un pattern
- Étapes de réalisation d'un pattern

Approche et méthode pédagogique

Approche structurée et pragmatique centrée sur la compréhension et l'application des design patterns. Chaque patron est présenté à l'aide d'un modèle générique en UML, indépendant du langage, suivi d'un exemple concret (Java ou C#) avec explications détaillées.Une approche permettant une compréhension claire, structurée et directement applicable des design patterns dans les projets réels.

Prérequis

- Bonne expérience en programmation orientée objet
- Maîtrise des concepts de conception objet (ex. SOLID)
- Connaissance des diagrammes UML (diagrammes de classes)
- Expérience dans un langage orienté objet (Java, C#, etc.)

Recommandations

- Expérience pratique en développement logiciel en équipe
- Familiarité avec les enjeux d'architecture et de conception d'applications
- Intérêt pour les bonnes pratiques de conception et la maintenabilité du code
- Capacité à analyser des problématiques de conception