Fiche du cours

60 h

Titre :

JAV100 - Programmation Java 1

Description :

Introduction pratique à Java (17/21 LTS) pour poser des bases solides : syntaxe, types, contrôle de flux, POO (classes/objets), collections & génériques, exceptions et entrées/sorties. On apprend à structurer un petit projet avec Maven/Gradle et à écrire ses premiers tests JUnit 5. Vous découvrirez aussi les bonnes pratiques (naming, immutabilité, gestion des erreurs) et l'usage d'un IDE moderne (IntelliJ/VS Code) pour gagner en efficacité. En fin de module, vous serez capable de livrer une petite application console propre, versionnée avec Git et prête à évoluer.

Objectifs :

- Comprendre la syntaxe Java et les types (primitifs/références), opérateurs et conversions.*
- Utiliser conditions/boucles, méthodes et surcharge.*
- Appliquer les bases POO : classes, objets, encapsulation, héritage, polymorphisme (intro).*
- Manipuler chaînes, collections (List, Set, Map) et génériques (bases).*
- Gérer les exceptions (try/catch/throws) et lire/écrire des fichiers (NIO.2).*
- Construire un projet simple avec Maven/Gradle et écrire des tests JUnit 5.

Chapitres :

1. Environnement & outils (JDK 17/21, IDE, Maven/Gradle), structure d'un projet*
2. Variables, types, opérateurs, immutabilité, var (local)*
3. Contrôle de flux : if/else, switch, for/while, break/continue*
4. Méthodes, portée, surcharge ; notions de packages*
5. POO I : classes, objets, constructeurs, this, encapsulation*
6. POO II (intro) : héritage, polymorphisme, override/final*
7. Chaînes & formatage (String, StringBuilder, format)*
8. Collections & génériques (bases) : List/Set/Map, itérations*
9. Exceptions : hiérarchie, checked vs unchecked, bonnes pratiques*
10. Fichiers (NIO.2) : chemins, lecture/écriture, encodage*
11. Tests (intro) : JUnit 5, assertions de base*
12. Mini-projet console : petite application structurée + tests essentiels

À la fin :
Vous saurez rédiger et organiser des programmes Java clairs, appliquer la POO, manipuler les collections, gérer exceptions et I/O, et livrer un petit projet outillé (Maven/Gradle + JUnit) prêt à évoluer. Vous utiliserez un IDE moderne (IntelliJ/VS Code) et Git pour versionner, déboguer et documenter proprement votre code. Vous aurez acquis les réflexes de tests et de qualité (structure de packages, lisibilité, gestion des erreurs) pour aborder sereinement JAV200.