FONDAMENTAUX : HTML, CSS, JS, PYTHON

Les fondamentaux de la programmation Java et POO

Acquérir les fondamentaux du langage Java et de la programmation orientée objet pour aborder Spring Boot dans de bonnes conditions. Syntaxe, POO, collections, exceptions et streams : les bases indispensables du développeur Java backend.

DURÉE	TARIF HT	NIVEAU	LANGUE	GROUPE	FORMAT
5j (35h)	**1200.00 €**	**Débutant**	**Français**	**3-12**	**Formation**

1 PUBLIC VISÉ

- Débutants en Java souhaitant apprendre le langage pour évoluer vers le développement backend en entreprise.
- Développeurs connaissant un autre langage (PHP, JavaScript, Python) souhaitant ajouter Java à leur profil.
- Personnes visant un poste de développeur backend Java et souhaitant préparer l'apprentissage de Spring Boot.
- Étudiants ou personnes en reconversion ciblant des environnements enterprise où Java est le standard.

2 PRÉREQUIS

Aucune expérience Java requise.

Des notions d'algorithmique (variables, conditions, boucles) dans n'importe quel langage sont fortement recommandées.

La formation Algorithmique : les bases est conseillée pour les débutants absolus. Savoir utiliser un ordinateur et un navigateur web.

3 OBJECTIFS PÉDAGOGIQUES

- Comprendre les spécificités du langage Java : typage statique, compilation, JVM
- Maîtriser la syntaxe Java : variables, types primitifs, conditions, boucles et tableaux
- Concevoir et utiliser des classes Java : encapsulation, constructeurs, méthodes
- Appliquer les principes de la POO : héritage, polymorphisme, classes abstraites et interfaces
- Utiliser les collections Java : List, Map, Set et leurs implémentations courantes
- Gérer les exceptions avec try, catch, finally et créer ses propres exceptions
- Découvrir les streams Java 8+ pour traiter des collections de manière fonctionnelle
- Comprendre les bases de la gestion des fichiers pour préparer l'accès aux données avec Spring Boot

4 PROGRAMME DÉTAILLÉ

Jour 1 (7h) – Découverte de Java et bases du langage

Module 1 – Pourquoi Java et mise en place de l'environnement (2h30)

Java dans le monde réel (1h)

- Java dans l'entreprise : banques, assurances, e-commerce, Android
- La JVM : écrire une fois, exécuter partout — ce que cela signifie concrètement
- Java vs JavaScript : deux langages sans rapport malgré le nom
- Le cycle de vie d'un programme Java : source .java → bytecode .class → exécution JVM
- Java et Spring Boot : pourquoi maîtriser Java avant d'aborder le framework

Installation et premier programme (1h30)

- Installer le JDK 21 (LTS) et IntelliJ IDEA Community
- Créer un projet Java dans IntelliJ : structure src/, Main.java
- La méthode main : point d'entrée d'un programme Java
- System.out.println : afficher des données dans la console
- Compiler et exécuter depuis IntelliJ et depuis le terminal
- Lire un message d'erreur du compilateur Java
- *Cas pratique : programme qui affiche des informations sur une personne*

Module 2 – Types primitifs, variables et opérateurs (4h30)

Variables et types primitifs (2h)

- Les huit types primitifs : int, long, double, float, boolean, char, byte, short
- Déclarer et initialiser une variable : typage statique obligatoire
- L'inférence de type avec var (Java 10+)
- Les constantes avec final
- Conversion de types : casting implicite et explicite
- La classe String : méthodes essentielles, concaténation, formatted()
- *Cas pratique : calculateur de tarif avec variables typées*

Opérateurs et expressions (1h30)

- Opérateurs arithmétiques, de comparaison et logiques
- L'opérateur ternaire
- Priorité des opérateurs
- Autoboxing : Integer, Double, Boolean — primitifs et leurs wrappers
- *Cas pratique : calcul de TVA et remise avec affichage formaté*

Atelier de consolidation (1h)

- Exercices libres sur les types et les expressions
- Correction collective et questions/réponses

Jour 2 (7h) – Structures de contrôle et introduction à la POO

Module 3 – Conditions, boucles et tableaux (3h30)

Structures de contrôle (1h15)

- if, else if, else : syntaxe et bonnes pratiques
- switch classique et switch expression Java 14+ avec les flèches
- *Cas pratique : système de classification selon un score*

Boucles (1h15)

- for : boucle à compteur
- while et do...while : boucle conditionnelle
- for-each : parcourir un tableau ou une collection
- break et continue : contrôler le flux
- *Cas pratique : table de multiplication et recherche dans un tableau*

Tableaux (1h)

- Déclarer et initialiser un tableau : syntaxe Java
- Tableaux multidimensionnels
- La classe Arrays : sort, copyOf, fill, toString
- Les limites des tableaux Java : taille fixe
- *Cas pratique : gestion d'une liste de notes avec calcul de moyenne*

Module 4 – Concevoir une classe Java (3h30)

Déclarer et instancier (1h45)

- Déclarer une classe : champs, constructeur, méthodes
- Instancier un objet avec new et accéder à ses membres
- Le mot-clé this : référencer l'instance courante
- Encapsulation : private sur les champs, getters et setters
- Constructeurs surchargés : plusieurs façons d'instancier un objet
- Les records Java 16+ : classes immuables concises pour les données
- *Cas pratique : classe Produit avec validation dans les setters*

Méthodes avancées et classes utilitaires (1h45)

- Surcharge de méthodes : même nom, paramètres différents
- Méthodes et champs statiques : appartenir à la classe plutôt qu'à l'instance
- toString() : représentation textuelle d'un objet
- equals() et hashCode() : comparer des objets correctement
- La classe Math et ses méthodes utilitaires
- *Cas pratique : classe Panier avec calcul du total et affichage formaté*

Jour 3 (7h) – Héritage, polymorphisme et abstractions

Module 5 – Héritage (3h)

Étendre une classe (1h30)

- Étendre une classe avec extends
- Appeler le constructeur parent avec super()
- Surcharger une méthode avec @Override
- Le mot-clé final : interdire l'héritage ou la surcharge
- La classe Object : mère de toutes les classes Java
- *Cas pratique : hiérarchie de classes pour des types de comptes bancaires*

Polymorphisme (1h30)

- Référence de type parent, objet de type enfant
- Dispatch dynamique : la bonne méthode est appelée à l'exécution
- instanceof et le pattern matching Java 16+ : if (obj instanceof String s)
- *Cas pratique : système de facturation avec comportement différent par type de client*

Module 6 – Classes abstraites et interfaces (4h)

Classes abstraites (2h)

- Déclarer une classe abstraite avec abstract
- Méthodes abstraites : forcer l'implémentation dans les sous-classes
- Quand utiliser une classe abstraite vs une classe concrète
- *Cas pratique : classe abstraite Forme avec sous-classes Cercle, Rectangle, Triangle*
- Atelier : diagramme UML simplifié de la hiérarchie, mise en commun

Interfaces (2h)

- Déclarer une interface et l'implémenter avec implements
- Implémenter plusieurs interfaces : résoudre l'absence d'héritage multiple
- Les méthodes default dans les interfaces Java 8+
- Interface vs classe abstraite : la règle de décision
- Les interfaces fonctionnelles : préparer les lambdas et les streams
- *Cas pratique : interface Exportable et Comparable pour les entités métier*

Jour 4 (7h) – Collections, exceptions et streams

Module 7 – Collections Java (2h30)

List et Set (1h15)

- Le framework Collections : vue d'ensemble et hiérarchie
- ArrayList : liste dynamique indexée, cas d'usage
- LinkedList : liste chaînée, quand la préférer
- HashSet et LinkedHashSet : collection sans doublons
- Itérer avec for-each et iterator
- *Cas pratique : gestion d'une liste de produits avec ajout, suppression et recherche*

Map (1h15)

- HashMap : dictionnaire clé/valeur non ordonné

LinkedHashMap et TreeMap : ordre d'insertion et ordre naturel

- Méthodes essentielles : put, get, containsKey, entrySet, getOrDefault
- Itérer sur une Map avec entrySet
- *Cas pratique : compteur de mots et index de produits par catégorie*

Module 8 – Exceptions et gestion des erreurs (2h)

- La hiérarchie des exceptions : Throwable, Error, Exception, RuntimeException
- Checked vs unchecked exceptions : la distinction fondamentale
- try, catch, finally : intercepter et traiter les exceptions
- try-with-resources : fermer automatiquement les ressources
- Lancer une exception avec throw et déclarer avec throws
- Créer ses propres exceptions métier
- *Cas pratique : service de validation avec exceptions métier personnalisées*

Module 9 – Lambdas et streams Java 8+ (2h30)

Lambdas et interfaces fonctionnelles (1h)

- Les expressions lambda : syntaxe et cas d'usage
- Les interfaces fonctionnelles : Predicate, Function, Consumer, Supplier
- Les method references : Class::method

L'API Stream (1h30)

- Créer un stream depuis une collection ou un tableau
- Opérations intermédiaires : filter, map, sorted, distinct, limit
- Opérations terminales : collect, forEach, count, reduce, findFirst
- Collectors : toList, toMap, groupingBy
- *Cas pratique : filtrer, trier et regrouper une liste de produits avec les streams*

Jour 5 (7h) – Projet de synthèse et ouverture Spring Boot

Module 10 – Projet de synthèse (4h30)

Lancement et réalisation (3h)

- Cahier des charges : application console de gestion de stock ou de bibliothèque
- Fonctionnalités à implémenter :
  - Hiérarchie de classes avec héritage et interfaces
  - Collections pour stocker les données en mémoire
  - Gestion des exceptions métier personnalisées
  - Streams pour les recherches et les statistiques
  - Menu interactif en console avec Scanner

Revue de code collective (1h30)

- Présentation courte de chaque binôme ou participant
- Architecture POO : cohérence des classes, respect de l'encapsulation

Utilisation des collections et qualité des exceptions

- Retour formateur individualisé sur le projet rendu

Module 11 – Ouverture vers Spring Boot (2h30)

Du Java pur au framework (1h15)

- Comment les concepts vus s'appliquent dans Spring Boot
- Les beans et le conteneur IoC : @Component, @Service, @Repository
- L'injection de dépendances : @Autowired vs constructeur
- Les repositories et les entités : le lien avec la POO Java

Mise en perspective et suite du parcours (1h15)

- Démonstration live : transformer la classe Produit en entité Spring Boot JPA
- Les outils à maîtriser ensuite : Maven/Gradle, Spring Data, REST controllers
- Ressources recommandées pour continuer : documentation officielle, exercices pratiques
- Questions/réponses libres et bilan de formation

5 COMPÉTENCES VISÉES

- Écrire des programmes Java orientés objet lisibles et bien structurés
- Concevoir une hiérarchie de classes adaptée à un problème métier
- Manipuler les structures de données Java : List, Map, Set
- Gérer les erreurs et exceptions de manière robuste
- Utiliser les streams et les lambdas pour traiter des collections
- Lire la documentation Java et comprendre les messages d'erreur du compilateur

6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel ou distanciel (visioconférence). Le formateur alterne entre méthode démonstrative (live coding commenté avec retour immédiat du compilateur Java), méthode interrogative (analyse d'erreurs de compilation et de conception orientée objet) et méthode active (exercices pratiques et mini-projets par module). L'accent est mis sur la compréhension des mécanismes du langage Java et des fondamentaux POO indispensables à l'apprentissage de Spring Boot.

7 MOYENS PÉDAGOGIQUES ET TECHNIQUES

- Support de cours numérique mis à disposition des apprenants
- Dépôt GitHub de démonstration avec exercices et corrections par module
- Environnement de développement : IntelliJ IDEA Community + JDK 21
- Pour le distanciel : visioconférence (Zoom ou équivalent), partage d'écran, chat en direct
- Accès à la plateforme pédagogique LaPolaris (supports, ressources, émargement)

8 MODALITÉS D'ÉVALUATION

- En cours de formation : exercices pratiques et mini-projets corrigés à chaque module
- En fin de formation : réalisation d'une application Java console orientée objet complète
- Questionnaire d'auto-évaluation des acquis en fin de parcours

9 CRITÈRES D'ÉVALUATION

- Écriture de code Java compilable et lisible sans erreurs de syntaxe
- Conception correcte d'une hiérarchie de classes avec encapsulation et héritage
- Utilisation adaptée des collections Java selon le contexte
- Gestion robuste des exceptions avec des messages d'erreur explicites
- Utilisation des streams pour remplacer les boucles sur les collections

10 MODALITÉS DE VALIDATION

Attestation de fin de formation délivrée à l'issue du parcours, conditionnée à une assiduité d'au moins 80 % et à la réalisation du projet de synthèse. L'attestation précise les objectifs atteints et les compétences acquises.

11 SUIVI ET ACCOMPAGNEMENT

- Feuilles d'émargement signées par demi-journée (présentiel) ou émargement numérique (distanciel)
- Traçabilité des activités pédagogiques réalisées
- Attestation d'assiduité délivrée en fin de formation
- Suivi individuel via les exercices corrigés et le projet de synthèse

12 CONDITIONS D'ACCÈS

Formation accessible sur inscription directe, sans prérequis administratif particulier. Le financement peut être pris en charge par l'employeur dans le cadre d'un plan de développement des compétences, ou en autofinancement avec possibilité de paiement en plusieurs fois.

13 DÉLAIS D'ACCÈS

Inscription possible jusqu'à **5 jours ouvrés** avant le début de la session. Pour toute demande urgente, nous contacter directement.

ACCESSIBILITÉ · HANDICAP

Nos formations sont accessibles aux personnes en situation de handicap. Pour toute situation nécessitant un aménagement (matériel, temporel ou pédagogique), nous vous invitons à nous contacter avant l'inscription afin d'étudier les adaptations possibles. Référent handicap : contact@lapolaris.fr

LaPolaris

TÉL. +33762584798

EMAIL contact@lapolaris.fr

WEB lapolaris.fr