DAWAN

Plan de la formation

www.dawan.fr

Formation Java : Initiation + Approfondissement (Nouvelle version)

Formation éligible au CPF, contactez-nous au 02/318.50.01

Durée :	5 jours (35 heures)
Tarifs inter-entreprise :	2 375,00 € HT (standard) 1 900,00 € HT (remisé)
Public :	Développeurs, intégrateurs, analystes
Pré-requis :	Maîtriser les bases de l'algorithmique (variables, conditions, boucles) - Avoir déjà pratiqué un langage de programmation (type C, C#, JavaScript, Python, PHP...) est un plus - Être à l'aise avec l'utilisation d'un poste de travail (Windows, macOS ou Linux).
Objectifs :	Installer et configurer un environnement de développement Java (JDK, IDE) et créer un projet - Écrire du code Java en utilisant les instructions de base du langage (variables, conditions, boucles, méthodes) - Concevoir et implémenter des classes et objets en respectant les principes d'encapsulation - Mettre en œuvre l'héritage et le polymorphisme pour structurer un modèle objet évolutif - Gérer les erreurs d'exécution avec le mécanisme d'exceptions afin de sécuriser les traitements - Manipuler des collections simples (List, Set, Map) et l'API de dates pour gérer des données en mémoire - Construire une petite application console complète en Java, de la modélisation à la réalisation - Approfondir l'utilisation des collections, des comparateurs, des lambdas et des premiers streams pour manipuler des ensembles de données - Lire et écrire des fichiers texte avec les API modernes de Java et externaliser la configuration de l'application - Configurer et utiliser JDBC pour se connecter à une base de données relationnelle et réaliser des opérations CRUD - Gérer les erreurs et transactions liées aux accès aux données afin de garantir l'intégrité des informations - Organiser le code en couches (présentation console, services métier, DAO) pour améliorer la maintenabilité - Finaliser une application console complète en Java intégrant POO, collections, fichiers et persistance en base.

Modalités pédagogiques, techniques et d’encadrement :	• Formation synchrone en présentiel et distanciel. • Méthodologie basée sur l’Active Learning : 75 % de pratique minimum. • Un PC par participant en présentiel, possibilité de mettre à disposition en bureau à distance un PC et l’environnement adéquat. • Un formateur expert.
Modalités d’évaluation :	• Définition des besoins et attentes des apprenants en amont de la formation. • Auto-positionnement à l'entrée et la sortie de la formation. • Suivi continu par les formateurs durant les ateliers pratiques. • Évaluation à chaud de l'adéquation au besoin professionnel des apprenants le dernier jour de formation.
Sanction :	Attestation de fin de formation mentionnant le résultat des acquis
Référence :	JAV102781-F
Demandeurs d’emploi:	Des entreprises recrutent des demandeurs d'emploi qui ont suivi ce cours dans le cadre d'une POEI, contactez-nous au 09.72.37.73.73 pour plus d'informations.
Note de satisfaction des participants:	4,59 / 5
Contacts :	commercial@dawan.fr - 09 72 37 73 73
Modalités d’accès :	Possibilité de faire un devis en ligne (www.dawan.fr, moncompteformation.gouv.fr, maformation.fr, etc.) ou en appelant au standard.
Délais d'accès :	Variable selon le type de financement.
Accessibilité :	Si vous êtes en situation de handicap, nous sommes en mesure de vous accueillir, n'hésitez pas à nous contacter à referenthandicap@dawan.fr, nous étudierons ensemble vos besoins

Découvrir la plateforme Java et l’environnement de développement

Comprendre le rôle du JDK, de la JVM et du bytecode
Identifier les éditions Java : Java SE, Java EE, Java ME et leurs usages
Installer et configurer un environnement de développement (JDK, IDE type Eclipse ou IntelliJ)
Structurer un projet Java : packages, organisation des sources, exécution d’une classe main

Compiler et exécuter une application en ligne de commande et depuis l'IDE
Découvrir l'empaquetage simple en jar exécutable
Atelier fil rouge : Créer un premier projet Java et une application console "Hello + mini-menu"

Maîtriser les bases du langage Java

Déclarer et utiliser variables, constantes et types primitifs
Utiliser les opérateurs arithmétiques, logiques et de comparaison
Mettre en œuvre les structures conditionnelles : if/else, switch, opérateur ternaire
Réaliser des traitements répétitifs avec les boucles for, while, do...while
Manipuler des tableaux unidimensionnels et multidimensionnels
Définir des méthodes : paramètres, valeurs de retour, portée des variables
Factoriser le code par la décomposition en sous-méthodes
Adopter des conventions de nommage et de documentation (JavaDoc) pour améliorer la lisibilité
Atelier fil rouge : Implémenter les premières fonctionnalités du menu console (saisies, calculs, affichages)

Appliquer les principes de la programmation orientée objet

Comprendre les notions de classe, objet, état et comportement
Créer des classes métier simples (POJO / JavaBean) avec attributs et méthodes
Mettre en œuvre l'encapsulation : niveaux de visibilité, getters et setters
Concevoir des constructeurs et les surcharger pour simplifier l'initialisation
Distinguer membres d'instance et membres de classe (static)
Introduire la modélisation avec un diagramme de classes UML simple pour préparer le code
Instancier des objets, comprendre la notion de référence et le rôle du garbage collector
Atelier fil rouge : Modéliser puis coder les premières classes métier de l'application (clients, produits, etc.)

Mettre en œuvre l'héritage et le polymorphisme

Spécialiser des classes à l'aide de l'héritage
Redéfinir des méthodes (override) et utiliser super pour réutiliser le comportement parent
Comprendre le polymorphisme et le typage dynamique des références
Utiliser des classes abstraites pour factoriser des comportements communs
Découvrir les interfaces pour définir des contrats (comportements attendus)

Illustrer l’intérêt de la POO pour faire évoluer l’application sans tout réécrire
Atelier fil rouge : Étendre le modèle objet avec une hiérarchie de classes et tester le polymorphisme dans l’application

Gérer les exceptions et sécuriser le code

Différencier erreurs, exceptions contrôlées et non contrôlées
Comprendre la propagation d’une exception dans la pile d’appels
Intercepter les erreurs avec try/catch/finally pour éviter les arrêts brutaux
Utiliser try-with-resources pour gérer automatiquement certaines ressources
Lever et propager des exceptions pour signaler les problèmes métier
Créer des exceptions applicatives personnalisées pour un diagnostic plus précis
Mettre en place des messages d’erreur clairs côté interface console
Atelier fil rouge : Rendre l’application robuste face aux saisies invalides et aux traitements anormaux

Découvrir les collections et l’API de dates

Comprendre les limites des tableaux pour la gestion de listes dynamiques
Découvrir les principales interfaces de collections : List, Set, Map
Utiliser quelques implémentations courantes (ArrayList, HashSet, HashMap) sans rentrer dans les détails internes
Parcourir et manipuler les collections avec les boucles et les itérateurs
Introduire la notion de generics pour sécuriser les types manipulés
Découvrir l’API java.time : LocalDate, LocalDateTime, durée et formatage simple
Atelier fil rouge : Gérer une collection d’objets métier (ajout, suppression, recherche simple, affichage formaté)

Approfondir les collections, lambdas et premiers streams

Revenir sur le choix des collections selon le besoin fonctionnel
Trier des collections à l’aide de Comparable et Comparator
Introduire les expressions lambda pour écrire des traitements compacts sur les objets
Découvrir les interfaces fonctionnelles courantes (Runnable, Comparator, Predicate, etc.)
Utiliser l’API Stream pour filtrer, transformer et agréger des données de manière lisible
Mettre en pratique quelques opérations courantes : filter, map, sorted, collect
Atelier fil rouge : Implémenter des recherches, tris et statistiques sur les collections de l’application

Lire et écrire des fichiers, externaliser la configuration

Utiliser l’API NIO.2 (java.nio.file) pour lire et écrire des fichiers texte
Gérer les chemins, répertoires, création et suppression de fichiers
Mettre en place une logique d’import/export simple de données au format texte (CSV ou équivalent)
Externaliser des paramètres dans des fichiers de propriétés (config applicative)
Appliquer une gestion d’erreurs cohérente sur les opérations d’entrée/sortie
Atelier fil rouge : Ajouter à l’application des fonctions d’import/export de données et un fichier de configuration

Accéder à une base de données relationnelle avec JDBC

Comprendre l’intérêt de la persistance en base de données pour une application Java
Configurer un pilote JDBC et établir une connexion à une base de données (type PostgreSQL, MySQL, etc.)
Écrire et exécuter des requêtes SQL simples : SELECT, INSERT, UPDATE, DELETE
Traiter les résultats avec ResultSet et mapper les lignes vers des objets métier
Utiliser les PreparedStatement pour paramétrer les requêtes et sécuriser les entrées
Gérer les transactions : commit, rollback, gestion des erreurs liées à la base
Atelier fil rouge : Persister les principales entités de l’application en base (création, consultation, mise à jour, suppression)

Structurer la couche d’accès aux données et l’architecture de l’application

Organiser le code en couches simples : présentation console, services métier, accès aux données
Mettre en place le pattern DAO pour isoler l’accès à la base
Séparer les responsabilités pour faciliter les tests et l’évolution de l’application
Introduire la notion de tests simples (exécution de scénarios via main, jeux de données) pour valider les traitements
Préparer la transition vers des frameworks plus avancés (JPA, Spring, etc.) sans les utiliser encore
Atelier fil rouge : Réorganiser le code existant en couches et factoriser l’accès aux données dans des DAO dédiés

Consolider les acquis par un projet complet en ligne de commande

Reprendre le cahier des charges global de l’application console (gestion d’un mini SI : stock, clients, commandes, etc.)
Valider l’utilisation cohérente de la POO, des collections, des exceptions et de l’accès aux données
Compléter les fonctionnalités manquantes, améliorer les messages d’erreur et les contrôles de cohérence

Mettre en place quelques scénarios de test de bout en bout
Faire le lien avec les compétences évaluées dans le cadre de la certification “Développer en langage Java” (sans imposer l’examen)
Atelier fil rouge : Finaliser l’application complète, exécuter des scénarios réalistes et présenter le travail réalisé