Formation : Java, les fondamentaux de la programmation

Cours pratique - 5j - 35h00 - Réf. LJB
Prix : 3070 CHF H.T.

NEW

À l'issue de la formation, le participant sera capable d'utiliser le langage Java et les technologies associées pour créer une application.

Objectifs pédagogiques

À l'issue de la formation, le participant sera en mesure de :

- Connaître les concepts de base du langage Java et maîtriser sa syntaxe
- Utiliser les bibliothèques et API
- Comprendre les concepts de la programmation orientée objet en Java
- Créer une application en Java

Public concerné

Développeurs, chargés de développement d'applications informatiques, chefs de projet proches du développement…

Prérequis

Connaître les principes de la programmation orientée objet et disposer d'une expérience sur un langage de programmation dans le développement d'applications.

Méthodes et moyens pédagogiques

Quiz, jeux en binômes ou en groupe, moments d'échanges, mise en situation, travaux pratiques intensifs, environnement de développement intégré.

Méthodes pédagogiques

60% pratique – 40% théorie. Pour optimiser le parcours d'apprentissage, des modules e-learning peuvent être fournis avant et après la session présentielle ou la classe virtuelle, sur simple demande du participant.

PARTICIPANTS

Développeurs, chargés de développement d'applications informatiques, chefs de projet proches du développement…

PRÉREQUIS

Connaître les principes de la programmation orientée objet et disposer d'une expérience sur un langage de programmation dans le développement d'applications.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

Modalités d'évaluation

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…

Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

Programme de la formation

1 Algorithmique – Raisonner avant de concevoir - OPTION digital learning préformation

- Introduction à l'algorithmique.
- Les instructions de base en pseudo-code.

Activités digitales

Dans cette formation en ligne, vous apprendrez à raisonner avant de concevoir un programme en découvrant les bases de l'algorithmique. Vous étudierez notamment les instructions fondamentales en pseudo-code.

2 Les techniques objet

- Les principes généraux de la modélisation et de la programmation objet.
- L'abstraction et l'encapsulation : les interfaces.
- Les différentes formes d'héritage, le polymorphisme.
- Introduction à la modélisation UML : le modèle statique, le modèle dynamique, le modèle de coopération, les scénarios.

Travaux pratiques

Spécification UML d'une étude de cas qui sera l'un des fils directeurs des exercices suivants.

3 Les constructions de base du langage

- Les variables : déclaration et typage.
- La définition des champs et des méthodes.
- Les expressions et instructions de contrôle.
- Les tableaux et types énumérés, l'autoboxing.
- Les unités de compilation et packages.

Travaux pratiques

Suite d'exercices simples permettant la prise en main de l'environnement de développement.

4 Application des textes et responsabilités

- Les imports statiques.
- Les entrées/sorties clavier.
- L'API java.time et conversion de dates.

Travaux pratiques

Réalisation d'un programme simple et utilisation des packages.

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les formations pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque formation ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le participant a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Pour toute question ou besoin relatif à l'accessibilité, vous pouvez joindre notre équipe PSH par e-mail à l'adresse psh-accueil@orsys.fr.

5 La définition et l'instanciation des classes

- Les classes et les objets.
- Les champs, les méthodes, les constructeurs.
- L'autoréférence et les champs/méthodes statiques.
- Les méthodes à nombre variable d'arguments.
- Les aspects méthodologiques : la conception des classes.

Travaux pratiques
Programmation de l'étude de cas en sous-groupes.

6 L'héritage (partie 1)

- Les différentes formes d'héritage : l'extension et l'implémentation.
- Les interfaces et l'implémentation des interfaces.
- Le polymorphisme et sa mise en œuvre.

Travaux pratiques
Conception et construction d'une hiérarchie de classes simples.

7 L'héritage (partie 2)

- L'extension, la définition des classes dérivées, les constructeurs, les références.
- La construction de hiérarchies de classes, la factorisation de code : les classes abstraites.
- L'utilisation simultanée de l'implémentation et de l'extension.

Travaux pratiques
Mise en place du polymorphisme et de la généricité dans l'étude de cas.
Construction de hiérarchies de classes et d'interfaces.

8 Les exceptions

- Les blocs try, la génération des exceptions.
- L'algorithme de sélection du catch().
- Les exceptions contrôlées et non contrôlées.
- Utilisation du bloc finally.

Travaux pratiques
Introduction des exceptions dans l'étude de cas.

9 Collections et généricité

- Notion de généricité et intérêt de la généricité.
- L'interface collection et types de listes.
- Les maps.

Travaux pratiques
Utilisation d'une classe générique et mise en œuvre des listes et map.

10 La programmation fonctionnelle

- Notion d'interface fonctionnelle.
- API java.util.fonction, les quatre catégories d'interfaces fonctionnelles.
- Les collections, les méthodes forEach et removeIf.
- Syntaxe et utilisation des expressions lambda.

Travaux pratiques
Utilisation d'expressions lambda avec une interface fonctionnelle. Application dans les listes et collections.

11 Les streams

- Relation avec la programmation fonctionnelle.
- Les opérateurs essentiels : filter, map, reduce.
- Notion d'opérations terminales et intermédiaires.
- Simplification d'algorithmes.

Travaux pratiques
Application des streams pour faire des traitements sur une collection.

12 Introduction à JDBC

- Principe et intérêt de Java Database Connectivity (JDBC).
- Architecture JDBC et pilotes.
- Configuration de l'environnement.

Travaux pratiques
Installation et configuration d'un pilote JDBC. Première connexion à une base de données.

13 Connexions et requêtes de base

- Notion de Connection, de Driver.
- Création et fermeture de connexions.
- Gestion des ressources.

Travaux pratiques
Établissement de connexions sécurisées.

14 Statement et ResultSet

- Notion de Statement et de ResultSet.
- Exécution de requêtes SELECT, INSERT, UPDATE, DELETE.
- Parcours et traitement des résultats.

Travaux pratiques
Mise en œuvre de requêtes basiques avec JDBC.

15 PreparedStatement et bonnes pratiques

- Statement et PreparedStatement, différences importantes.
- Prévention des injections SQL.
- Optimisation des performances.

Travaux pratiques
Conversion de Statement vers PreparedStatement. Implémentation de requêtes paramétrées.

16 Gestion des transactions

- Notion de transactions et bonnes pratiques.
- Commit et rollback.
- Gestion des erreurs dans les transactions.

Travaux pratiques
Mise en œuvre de transactions complexes.

17 Les nouveautés Java (versions récentes)

- Évolution de Java depuis Java 8.
- Les nouvelles API et fonctionnalités.
- Records, pattern matching, switch expressions.
- Text Blocks et API améliorées.
- Virtual Threads et autres améliorations.

Travaux pratiques
Refactoring d'ancien code avec les nouvelles fonctionnalités. Exercices pratiques sur les records et pattern matching. Expérimentation avec les text blocks.

18 Outils de débogage et monitoring

- Utilisation des outils de débogage IDE.
- Techniques de débogage avancées.
- Logging et monitoring.
- Bonnes pratiques de développement.

Travaux pratiques
Session de débogage pratique sur l'étude de cas.

19 Optimisation et performance (1/2)

- Profiling et analyse de performance.
- Gestion mémoire et Garbage Collector.

Travaux pratiques
Analyse de performance sur l'étude de cas.

20 Optimisation et performance (2/2)

- Optimisations courantes (arguments JVM).

Travaux pratiques
Mise en place d'optimisations.

21 Tests et qualité du code

- Stratégies de test modernes (TDD, BDD).
- JUnit 5 et ses nouvelles fonctionnalités.
- Tests paramétrés, tests d'intégration.
- Mocking avancé avec Mockito.
- Couverture de code et analyse statique.
- Tests de performance et de charge.

Travaux pratiques
Mise en place d'une suite de tests complète pour l'étude de cas. Configuration d'outils d'analyse de code.

22 Projet intégrateur (partie 1)

- Analyse des besoins et conception.
- Architecture de l'application.
- Mise en place de la structure du projet.

Travaux pratiques
Conception de l'application finale.

23 Projet intégrateur (partie 2)

- Implémentation des fonctionnalités core.
- Intégration base de données.
- Gestion des exceptions.

Travaux pratiques
Intégration des concepts dans l'application finale.

24 Projet intégrateur (partie 3)

- Finalisation du développement.
- Tests et débogage.
- Documentation.

Travaux pratiques
Tests et validation de l'application.

25 Junit, maîtriser les tests unitaires en Java — OPTION digital learning

post-formation

- Introduction à JUnit et configuration.
- Comprendre et utiliser les annotations Junit.
- Structuration des tests et bonnes pratiques.
- Tests avancés avec Junit.
- Approfondir la maîtrise de JUnit.

Activités digitales
Dans cette formation en ligne, vous découvrirez comment maîtriser JUnit 5 pour écrire, organiser et exécuter des tests unitaires efficaces en Java. Après une introduction aux concepts fondamentaux, vous apprendrez à structurer les tests, utiliser les annotations clés, gérer les exceptions, créer des tests paramétrés et des suites de tests, ainsi qu'à intégrer JUnit avec Maven ou Gradle. Chaque notion sera illustrée par des exemples pratiques comme le test d'une calculatrice ou d'un inventaire.

Dates et lieux

CLASSE À DISTANCE
2026 : 23 mars, 15 juin, 28 sep., 7 déc.