

Flutter – Développement d'applications multiplateformes pour mobiles

3 jours (21 heures)

Perfectionnement

Présentiel ou classe à distance

à partir de

1905 € HT par personne (TVA en vigueur) **Repas inclus en présentiel**

Contenu mis à jour le 25/09/2026. Document téléchargé le 02/10/2026.

Objectifs

À l'issue de cette formation, vous serez capable de :

- Reconnaître les frameworks de développement mobile et la spécificité de Flutter
- Mettre en place un environnement de développement Flutter
- Concevoir l'architecture logicielle d'une application mobile multiplateforme
- Construire une interface responsive
- Développer avec Dart
- Utiliser GitHub Copilot et Gemini in Android Studio pour accélérer le développement Flutter / Dart.

Les compétences visées par cette formation font l'objet d'une évaluation.

À qui s'adresse cette formation ?

Développeurs, chefs de projets.

Prérequis

Avoir des connaissances de base en programmation.

Programme

1 Pendant votre formation

Flutter – Développement d'applications multiplateformes pour mobiles

Jour 1 – Matin

Introduction à Flutter

- Origine du produit
- Pourquoi le nouveau langage Dart ?
- Architecture d'une application Flutter
- Android Studio et IntelliJ
- Le "fast development" grâce au Hot Reload
- Mise en place du débogueur
- Material Design étendu pour iOS (Cupertino) et Android
- L'accès aux Sensors et fonctionnalités du smartphone
- Création d'un premier projet dans Android Studio
- Utilisation du simulateur
- Coder dans main.dart
- Installation et activation de GitHub Copilot dans Android Studio
- Présentation de Gemini in Android Studio : complétion de code, explication de code Flutter, génération de Widgets

Exemples de travaux pratiques (à titre indicatif)

- *Création d'une petite application en utilisant le scaffolding, lancement de l'émulateur, test et debug*
- *Utiliser Gemini in Android Studio pour expliquer la structure du projet généré (main.dart, pubspec.yaml, arborescence), puis poser des questions en langage naturel sur l'architecture Flutter*

Jour 1 – Après-midi

Présentation du langage Dart

- Les variables
- Constantes et final
- Les chaînes de caractères, les strings
- Les booléens
- Listes et Map
- Les conditions
- Les différentes sortes de boucle
- Les diverses sortes de fonction
- Les classes et objets avec Dart
- Constructeurs de classe
- Méthodes et variables

- Générer des fonctions Dart complexes (fonctions d'ordre supérieur, closures, async / await) via l'IA et analyser le résultat
- Utiliser l'IA pour expliquer et corriger des erreurs de typage Dart (null safety, late, required)
- Refactoring assisté : demander à l'IA de transformer un code impératif Dart en code idiomatique (cascade notation, spread operator)

Exemples de travaux pratiques (à titre indicatif)

- *Création d'une classe et d'instances en utilisant Dart, intégration au précédent exercice, test et debug*
- *Générer via GitHub Copilot une classe Dart avec constructeurs, méthodes et sérialisation JSON, puis l'intégrer à l'application existante et corriger les erreurs avec l'aide de l'IA*

Jour 2 – Matin

Environnement Dart et Flutter

- Les bibliothèques de Dart et ses constituants
- Gestion des listes et collections de Dart
- L'interface CLI de Flutter
- Architecture d'un projet Flutter sous Android Studio
- Utiliser l'IA pour identifier et comparer des packages pub.dev adaptés à un besoin fonctionnel précis (ex : state management, HTTP, storage)
- Générer la configuration pubspec.yaml (dépendances, assets, fonts) à partir d'une description de l'application cible

Les Widgets

- Structure et définition d'un widget
- L'objet Render, le rendering
- Construire des interfaces graphiques avec des widgets
- Les widgets dynamiques ou StatefulWidget
- Les widgets statiques ou StatelessWidget

Exemples de travaux pratiques (à titre indicatif)

- *Utilisation avancée de l'interface CLI de Flutter*
- *Développement d'une petite application utilisant les collections en Dart*
- *Utiliser l'IA pour générer les transformations de collections (filtrer, trier, regrouper une liste de données JSON) et les intégrer dans l'application Flutter*

Jour 2 – Après-midi

Les Widgets – Suite

- Les layouts
- Les trees (Elements et Widgets)
- Les piles
- Colonnes et lignes
- Les widgets de Cupertino (iOS)

- Passage d'informations entre widgets
- Les widgets de type "liste" et le scrolling associé
- Création de pages de saisie
- Héritage de widgets

Navigation et routing dans Flutter

- Définir des routes
- Navigation grâce aux routes
- Passage de données entre routes

Exemples de travaux pratiques (à titre indicatif)

- Création d'une application avec une interface graphique utilisant différents widgets, mise en oeuvre de la communication entre widgets parents et enfants
- Mise en oeuvre de routes afin de passer d'une page à une autre avec Flutter
- Générer le système de navigation de l'application (au moins 3 écrans et le passage de données) à l'aide de GitHub Copilot, puis tester et affiner les suggestions

Jour 3 – Matin

Architecture : les principaux design patterns dans Flutter

- Les patterns de gestion d'état système
- Choisir entre le pattern MVVM et BLoC
- Les autres importants patterns dans Flutter

Persistance des données

- Persistance avec SQLite
- Persistance avec l'ORM Moor (drift)
- Accès au système de fichiers
- Les "shared preferences"

Exemples de travaux pratiques (à titre indicatif)

- Demander à l'IA de comparer BLoC vs MVVM / Riverpod pour un cas d'usage précis et d'argumenter le choix
- Ajout de persistance de données en utilisant le système de fichiers, SQLite et Moor
- Générer la couche de persistance (schéma et DAO Drift) de l'application via GitHub Copilot, puis l'intégrer dans l'architecture BLoC ou MVVM mise en place précédemment

Jour 3 – Après-midi

Compléments

- Tests
- Internationalisation
- Déploiement

Exemples de travaux pratiques (à titre indicatif)

- Mettre en place une suite de tests unitaires et fonctionnels
- Générer les fichiers ARB d'internationalisation (FR / EN / ES) à partir des chaînes de l'application via ChatGPT
- Utiliser l'IA pour générer un guide de déploiement Play Store / App Store personnalisé au projet

Financement

Plusieurs dispositifs de financement sont accessibles via les OPCO (Opérateurs de Compétences), organismes agréés par le ministère du Travail dont le rôle est d'accompagner, collecter et gérer les contributions des entreprises au titre du financement de la formation professionnelle.

Pour plus d'information, une équipe de gestionnaires SKOLAE Formation spécialisée vous accompagne dans le choix de vos formations et la gestion administrative.

[Nous contacter pour en savoir plus →](#)

Modalités pédagogiques

Formation délivrée en présentiel ou distanciel* (blended-learning, e-learning, classe virtuelle, présentiel à distance).

Le formateur alterne entre méthode** démonstrative, interrogative et active (via des travaux pratiques et/ou des mises en situation).

Variables suivant les formations, les moyens pédagogiques mis en oeuvre sont :

- Ordinateurs Mac ou PC (sauf pour certains cours de l'offre Management), connexion internet fibre, tableau blanc ou paperboard, vidéoprojecteur ou écran tactile interactif (pour le distanciel)
- Environnements de formation installés sur les postes de travail ou en ligne
- Supports de cours et exercices

En cas de formation intra sur site externe à M2I, le client s'assure et s'engage également à avoir toutes les ressources matérielles pédagogiques nécessaires (équipements informatiques...) au bon déroulement de l'action de formation visée conformément aux prérequis indiqués dans le programme de formation communiqué.

* nous consulter pour la faisabilité en distanciel

** ratio variable selon le cours suivi

Accessibilité aux personnes en situation de handicap

Détails de la session

Du 7 décembre 2026 au 9 décembre 2026 (3 jours / 21 heures)

Nice

Du 7 décembre 2026 au 9 décembre 2026 Nice

- 7 décembre 2026 9h00 – 16h00
- 8 décembre 2026 9h00 – 16h00
- 9 décembre 2026 9h00 – 16h00