

React Native – Création d'applications mobiles multiplateformes

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

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

- Reconnaître les frameworks de développement mobile et la spécialité de React Native
- Mettre en place un environnement de développement React Native
- Concevoir l'architecture logicielle d'une application mobile multiplateforme
- Construire une interface utilisateur fluide et performante
- Utiliser les principaux composants et les API natives proposées par React Native
- Utiliser GitHub Copilot et ChatGPT pour accélérer le développement de composants, de hooks et de services React Native.

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

À qui s'adresse cette formation ?

Développeurs mobiles, développeurs Web, architectes, chefs de projets techniques.

Prérequis

Avoir une connaissance pratique de la programmation en JavaScript.

Programme

1 Pendant votre formation

React Native – Création d'applications mobiles multiplateformes

Jour 1 – Matin

Présentation de React Native

- Historique et positionnement dans l'écosystème mobile
- Différences entre React (Web) et React Native
- Fonctionnement : bridge natif, composants natifs vs Webview
- Présentation de l'environnement de développement :
 - Node.js, npm, Expo, React Native CLI
 - VS Code, Android Studio, Xcode (si possible)
- Installation et activation de GitHub Copilot dans Visual Studio Code
- Utiliser l'IA pour générer le scaffolding d'un projet React Native (Expo ou CLI) à partir d'une description fonctionnelle

Exemples de travaux pratiques (à titre indicatif)

- *Création d'une première application "Hello React Native !"*
- *Explication de l'architecture des fichiers*
- *Lancement sur un émulateur*
- *Utiliser GitHub Copilot pour générer le projet Expo, puis demander à l'IA d'expliquer chaque fichier de la structure et de proposer une organisation de dossiers adaptée à une application e-commerce (le projet fil rouge de la formation)*

Jour 1 – Après-midi

Les bases de la programmation React Native

- Le langage JSX
- Composants React : cycle de vie, Props, State et style
- Composants React Native
- Positionnement des composants
- Style et thème
- Les outils de débogage
- Gestion des formulaires
- Affichage des données dans des listes (FlatList)
- Générer un composant FlatList complet (item renderer, keyExtractor, séparateurs, état vide) à partir d'un modèle de données décrit en texte
- Utiliser l'IA pour générer un formulaire React Native avec validation (React Hook Form ou Formik) et feedback utilisateur

Exemples de travaux pratiques (à titre indicatif)

- Réaliser une application CRUD simple
- Générer via GitHub Copilot les composants de base de l'application e-commerce (ProductCard, ProductList avec FlatList, formulaire de recherche) et affiner le style avec l'aide de l'IA

Jour 2 – Matin

Navigation entre écrans

- Installation et configuration de React Navigation
- Stack Navigator, Tab Navigator et Drawer Navigator
- Passage de paramètres entre écrans
- Décrire le plan de navigation de l'application e-commerce en texte et générer la configuration React Navigation complète (Stack, Tab et Drawer) via Copilot
- Utiliser l'IA pour générer les types TypeScript de navigation (RootStackParamList, ScreenProps) pour chaque navigator
- Générer le passage de paramètres typés entre écrans (ex : ProductId de la liste vers le détail) avec gestion des cas null / undefined
- Demander à ChatGPT de comparer React Navigation vs Expo Router et recommander l'approche la plus adaptée au projet fil rouge

Gestion de l'état de l'application

- Gestion de l'état avec les Hooks : useState, useContext, useReducer...
- Présentation des autres alternatives de gestion de l'état

Exemples de travaux pratiques (à titre indicatif)

- Création de la première partie de l'application de commerce électronique : catalogue produits et page détails
- Générer via GitHub Copilot la structure de navigation complète de l'application e-commerce (Tab Navigator : Catalogue / Panier / Profil et Stack pour le détail produit) avec les types TypeScript correspondants

Jour 2 – Après-midi

Gestion de l'état de l'application – Suite

- Gestion de l'état d'une application complexe avec Redux
- Utilisation d'AsyncStorage pour la persistance des données locales
- Générer un store Redux Toolkit complet (slice, actions, selectors, thunk) pour le panier e-commerce à partir d'une description des fonctionnalités
- Utiliser GitHub Copilot pour générer les hooks personnalisés (useCart, useProducts, useAuth) encapsulant la logique Redux
- Générer la couche AsyncStorage (helpers typés : saveCart, loadCart, clearCart) pour la persistance du panier via l'IA
- Demander à ChatGPT de comparer Redux Toolkit vs Zustand vs Context API pour l'application e-commerce et justifier le choix

Communication avec des API REST

- Utilisation de fetch et axios pour la communication avec un serveur distant
- Appels GET, POST, PUT et DELETE
- Gestion des listes et composants dynamiques
- Générer un service API complet (axios instance, intercepteurs, gestion des erreurs, retry) via GitHub Copilot
- Utiliser l'IA pour générer les types TypeScript (interfaces Product, Order, Cart) à partir d'une réponse JSON d'API exemple
- Générer les hooks React Query ou SWR pour la gestion du cache et des états de chargement / erreur des appels REST

Exemples de travaux pratiques (à titre indicatif)

- Ajouter les fonctionnalités de gestion du panier et de persistance des données à l'application de commerce électronique
- Générer via Copilot le slice Redux du panier (ajout, suppression, quantité), le service API produits (GET /products, GET /products/:id) et la persistance AsyncStorage, puis intégrer l'ensemble dans l'application e-commerce

Jour 3 – Matin

Gestion de l'authentification

- Authentification via une API ou Firebase Authentication
- Gestion des tokens et navigation conditionnelle
- Générer l'écran de connexion React Native complet (formulaire, validation, loader, gestion des erreurs) via GitHub Copilot
- Utiliser l'IA pour générer le flow d'authentification JWT complet : login, stockage sécurisé du token (expo-secure-store), refresh token, déconnexion

Intégration des fonctionnalités natives

- Accès caméra (Expo Camera)
- Accès GPS
- Notifications locales
- Gestion des permissions (Expo Permissions)

Exemples de travaux pratiques (à titre indicatif)

- Intégration de l'authentification dans l'application de commerce électronique et ajout d'un composant de validation de la commande pour les utilisateurs authentifiés
- Générer via GitHub Copilot le service d'authentification Firebase, la navigation conditionnelle (AuthStack / AppStack) et l'écran de validation de commande réservé aux utilisateurs connectés
- Dans l'application de commerce électronique, intégration de l'envoi de notifications lorsqu'une commande est validée
- Générer via Copilot le service de notifications locales (expo-notifications) déclenché à la validation de commande, avec la gestion de la permission et la personnalisation du contenu de la notification

Jour 3 – Après-midi

Compilation et déploiement

- Compilation en APK (Android)
- Test sur appareils physiques
- Présentation des options de déploiement sur Google Play et App Store
- Utiliser ChatGPT pour générer une checklist de publication complète (Google Play et App Store) personnalisée à l'application e-commerce
- Générer les scripts EAS Build (Expo Application Services) pour la compilation Android (APK / AAB) et iOS via GitHub Copilot
- Utiliser l'IA pour générer la documentation technique complète du projet : README, architecture decision records, guide d'installation

Exemples de travaux pratiques (à titre indicatif)

- Finalisation de l'application de commerce électronique et déploiement sur un smartphone Android
- Générer via ChatGPT la checklist de déploiement et les descriptions store de l'application e-commerce, puis utiliser Copilot pour générer la configuration EAS Build et le README technique du 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 16 décembre 2026 au 18 décembre 2026 (3 jours / 21 heures)

Lille

Du 16 décembre 2026 au 18 décembre 2026

Lille

- 16 décembre 2026 ----- 9h00 – 16h00
- 17 décembre 2026 ----- 9h00 – 16h00
- 18 décembre 2026 ----- 9h00 – 16h00