DÉVELOPPEMENT FRONTEND

React : développer des interfaces modernes

Créer des interfaces web dynamiques avec React. Composants, hooks, gestion d'état, appels API : devenez opérationnel sur la bibliothèque la plus demandée du marché.

DURÉE	TARIF HT	NIVEAU	LANGUE	GROUPE	FORMAT
4j (28h)	**1200.00 €**	**Intermédiaire**	**Français**	**3-8**	**Formation**

1 PUBLIC VISÉ

- Développeurs JavaScript maîtrisant ES6+ souhaitant créer des interfaces web dynamiques et composants réutilisables avec React.
- Développeurs front-end ayant des bases en HTML/CSS/JS voulant adopter une bibliothèque moderne et professionnelle.
- Personnes ayant suivi la formation JavaScript fondamentaux ou la formation JavaScript avancé, ou disposant d'une expérience équivalente.

2 PRÉREQUIS

Bonne maîtrise de JavaScript ES6+ : fonctions fléchées, destructuring, modules, promesses, async/await. La formation JavaScript fondamentaux est recommandée. Savoir utiliser le terminal et VS Code.

3 OBJECTIFS PÉDAGOGIQUES

- Comprendre le paradigme par composants et savoir décomposer une interface en composants réutilisables
- Maîtriser les hooks essentiels : useState, useEffect, useRef, useMemo, useCallback
- Gérer l'état global d'une application avec Context API et des patterns adaptés
- Mettre en place une navigation multi-pages avec React Router v6
- Consommer une API REST depuis React avec gestion des états de chargement et d'erreur
- Gérer des formulaires contrôlés avec validation côté client
- Structurer un projet React professionnel et le déployer en production

4 PROGRAMME DÉTAILLÉ

Jour 1 (7h) – Fondamentaux React et composants

Module 1 – Prise en main de React (2h)

Pourquoi React ? (30min)

- Le problème que React résout : manipulation du DOM et réactivité
- Virtual DOM : principe et avantages
- React vs Vue vs Angular : positionnement et cas d'usage
- L'écosystème React : Vite, React Router, bibliothèques de composants

Mise en place de l'environnement (30min)

- Créer un projet React avec Vite
- Structure d'un projet React : src, composants, assets
- Les scripts npm : dev, build, preview
- Extensions VS Code recommandées : ES7 React Snippets, Prettier

JSX : JavaScript + HTML (1h)

- Comprendre JSX : ce que Babel transforme réellement
- Les règles JSX : un seul élément racine, className, camelCase
- Expressions JavaScript dans JSX : accolades, ternaires, short-circuit
- Rendu de listes avec .map() et l'importance de la clé key
- Cas pratique : construire une carte de profil utilisateur en JSX

Module 2 – Composants fonctionnels et props (2h)

Créer et composer des composants (1h)

- Composant fonctionnel : définition, nommage, export
- Importer et utiliser un composant dans un autre
- Quand créer un nouveau composant : la règle de responsabilité unique
- Arborescence de composants : parent, enfant, fratrie
- Cas pratique : décomposer une maquette en arbre de composants

Props : communication entre composants (1h)

- Passer des props à un composant enfant
- Destructuring des props, valeurs par défaut
- La prop children : composer des composants conteneurs
- Props et immuabilité : pourquoi un enfant ne modifie pas ses props
- Cas pratique : composant Card générique avec props de contenu

Module 3 – Premier hook : useState (3h)

Réactivité et état local (1h)

Qu'est-ce que l'état ? Différence entre variable et état React

- useState : syntaxe, valeur initiale, mise à jour
- React re-render : quand et pourquoi le composant se re-dessine
- Immutabilité : mettre à jour un objet ou un tableau dans l'état

Événements et interactivité (1h)

- Gestion des événements en React : onClick, onChange, onSubmit
- Passer des handlers comme props (lifting state up)
- Propagation des événements et preventDefault
- Cas pratique : compteur, toggle, liste dynamique

Rendu conditionnel (1h)

- Afficher ou masquer un composant selon l'état
- Ternaire vs short-circuit vs early return
- Afficher des états de chargement et d'erreur
- Cas pratique : modale contrôlée par l'état

Jour 2 (7h) – Hooks avancés et communication

Module 4 – useEffect et cycle de vie (3h)

Comprendre useEffect (1h30)

- Qu'est-ce qu'un effet de bord ?
- useEffect : syntaxe, tableau de dépendances, fonction de nettoyage
- Les quatre comportements selon le tableau de dépendances
- Les erreurs classiques : boucles infinies, dépendances manquantes
- La règle d'or : chaque effet fait une seule chose

useEffect en pratique (1h30)

- Appel API au montage du composant
- Synchronisation avec une valeur externe (titre de page, localStorage)
- Nettoyage : annuler un appel API, retirer un event listener
- Cas pratique : composant de recherche avec debounce

Module 5 – useRef, useMemo, useCallback (2h)

useRef : références sans re-render (1h)

- useRef pour accéder à un élément DOM : focus, scroll, animation
- useRef pour persister une valeur sans déclencher un re-render
- Cas pratique : champ de recherche auto-focus, compteur de renders

useMemo et useCallback : optimisation (1h)

- Quand React recalcule inutilement : identifier les re-renders superflus
- useMemo : mémoïser un calcul coûteux

useCallback : stabiliser une fonction passée en prop

- La règle : n'optimiser que si le problème est mesuré

Module 6 – Context API et état global (2h)

Le problème du prop drilling (30min)

- Quand l'état doit être partagé entre composants distants
- Le prop drilling : ses limites et quand s'en méfier

Context API (1h30)

- Créer un contexte avec createContext
- Fournir une valeur avec Provider
- Consommer le contexte avec useContext
- Créer un hook personnalisé pour encapsuler le contexte
- Cas pratique : thème clair/sombre, panier d'achat global

Jour 3 (7h) – Routing, API et formulaires

Module 7 – React Router v6 (3h)

Navigation SPA (1h)

- Qu'est-ce qu'une SPA ? Différence avec la navigation traditionnelle
- Installer et configurer React Router v6
- BrowserRouter, Routes, Route : structure de base
- Link et NavLink : navigation sans rechargement
- Paramètres d'URL : useParams

Routing avancé (1h)

- Routes imbriquées et Outlet
- Routes protégées : redirection si non authentifié
- useNavigate : navigation programmatique
- useSearchParams : gestion des paramètres de requête

Cas pratique (1h)

- Construire une application multi-pages : liste, détail, formulaire
- Breadcrumb dynamique basé sur la route active

Module 8 – Appels API avec fetch et axios (2h)

Intégration REST depuis React (1h)

- fetch vs axios : cas d'usage, gestion des erreurs, intercepteurs
- Pattern standard : loading / data / error dans l'état
- Annuler une requête avec AbortController
- Cas pratique : liste de ressources avec pagination

Créer un hook personnalisé useFetch (1h)

- Extraire la logique d'appel API dans un hook réutilisable
- Paramétrer le hook : URL, méthode, headers
- Gérer la mise en cache simple
- Cas pratique : useFetch utilisé sur plusieurs pages

Module 9 – Formulaires contrôlés (2h)

Formulaires React (1h)

- Composants contrôlés vs non contrôlés
- Gérer un formulaire avec useState : champs texte, select, checkbox
- Validation synchrone : messages d'erreur par champ
- Soumettre un formulaire et envoyer les données à une API

Cas pratique (1h)

- Formulaire d'inscription complet : validation, feedback, reset
- Gestion du double submit et des états de chargement

Jour 4 (7h) – Architecture, bonnes pratiques et projet de synthèse

Module 10 – Structurer un projet React professionnel (2h)

Architecture et organisation (1h)

- Organisation des dossiers : components, pages, hooks, services, utils
- Composants présentationnels vs composants conteneurs
- Créer une bibliothèque de composants UI réutilisables
- Conventions de nommage et cohérence du code

Bonnes pratiques React (1h)

- Les règles des hooks : appel conditionnel, boucles
- Éviter les re-renders inutiles : React.memo
- PropTypes : documenter et valider ses props
- Gérer les variables d'environnement dans Vite (.env)

Module 11 – Déploiement (1h)

- Build de production avec Vite : optimisation, tree-shaking, chunks
- Déployer sur Vercel : connexion GitHub, variables d'environnement
- Déployer sur Netlite : configuration des redirections pour SPA
- Vérifier les performances : Lighthouse, Core Web Vitals

Module 12 – Projet de synthèse (4h)

Réalisation d'une application React complète

- Cahier des charges : application de gestion de ressources (films, livres ou produits au choix)
- Fonctionnalités : navigation multi-pages avec React Router, consommation d'une API REST publique, état global avec Context API, formulaire contrôlé avec validation, déploiement sur Vercel ou Netlify
- Revue de code collective : architecture, découpage composants, gestion des erreurs
- Retour formateur individualisé sur le projet rendu

5 COMPÉTENCES VISÉES

- Concevoir et développer une application React from scratch avec une architecture composants claire
- Gérer l'état local et global d'une application de manière fiable et prévisible
- Maîtriser les effets de bord et le cycle de vie des composants fonctionnels
- Intégrer une API REST avec gestion robuste des erreurs et des états asynchrones
- Écrire des formulaires contrôlés avec validation et feedback utilisateur
- Déployer une application React sur une plateforme de production (Vercel ou Netlify)

6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel ou distanciel (visioconférence). Le formateur alterne entre méthode démonstrative (live coding commenté et progressif), méthode interrogative (analyse et critique de code existant) et méthode active (exercices pratiques et mini-projets par module). L'accent est mis sur la compréhension des mécanismes internes de React plutôt que sur la mémorisation de l'API.

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 : VS Code + Node.js + Vite + React DevTools
- 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 React complète avec routing, API et état global
- Questionnaire d'auto-évaluation des acquis en fin de parcours

9 CRITÈRES D'ÉVALUATION

- Découpage pertinent de l'interface en composants réutilisables et bien séparés
- Utilisation correcte des hooks essentiels (useState, useEffect, useRef) sans violation des règles des hooks
- Gestion fiable des appels API avec états de chargement et d'erreur explicites
- Navigation fonctionnelle avec React Router et routes protégées
- Qualité et lisibilité du code : nommage, organisation des fichiers, absence de logique dupliquée

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