Vue.js - Fonctionnalités avancées

Référence : **DEVU002** Durée : **2 jours** Certification : **Aucune**

CONNAISSANCES PREALABLES

• Avoir des connaissances sur Vue.js.

PROFIL DES STAGIAIRES

• Cette formation Vue.js s'adresse aux développeurs Vue.js souhaitant produire des applications Vue.js plus optimisées et qualitatives.

OBJECTIFS

• Décrire les aspects avancés du Vue.js. • Créer des composants plus réutilisables. • Développer une application plus optimisée.

CERTIFICATION PREPAREE

Aucune

METHODES PEDAGOGIQUES

- Mise à disposition d'un poste de travail par stagiaire
- Remise d'une documentation pédagogique numérique pendant le stage
- La formation est constituée d'apports théoriques, d'exercices pratiques, de réflexions et de retours d'expérience
- Le suivi de cette formation donne lieu à la signature d'une feuille d'émargement

FORMATEUR

Consultant-Formateur expert Développement

METHODE D'EVALUATION DES ACQUIS

- Auto-évaluation des acquis par le stagiaire via un questionnaire
- Attestation des compétences acquises envoyée au stagiaire
- Attestation de fin de stage adressée avec la facture

CONTENU DU COURS

Jour 1

Approfondissement de Vue Router

- Chargement dynamique des composants
- Le mode "historique"
- La configuration server-side
- Les paramètres d'URL
- Routes imbriquées / Nested routes
- Navigation par le code
- Redirection
- Gestion d'erreur
- Flash message
- La pagination
- Les gardes

Exemple de travaux pratiques (à titre indicatif)

- Mise en œuvre des routes Vue.js dans l'application

Les slots

- Définition et utilité des slots
- Injection de contenu dans un template
- Slots et composants génériques
- Slots et composants parents
- Props et slots
- La directive v-slot
- Les slots nommés
- La portée des slots
- Les slots à nom dynamique

Exemples de travaux pratiques (à titre indicatif)

- Utilisation des slots pour réaliser un composant dont le contenu dépend de sa position dans la page (composant générique)
- Ajout de chargement lazy pour des routes référençant des composants sur le projet existant

Les composants asynchrones

- Utilité de composants asynchrones
- Chargement des composants à la demande
- La fonction "defineAsyncComponent"
- Retour sur les promises
- Gestion des erreurs et de l'état

Exemple de travaux pratiques (à titre indicatif)

- Intégration d'appels de composants de manière asynchrones dans l'application écrite

Jour 2

Vuex et Pinia

- Définition du modèle de gestion d'état (state management pattern)
- Vuex ou Pinia ?
- Un seul stockage par application
- Exemple de stockage le plus élémentaires
- Gestion de l'état Vuex dans les composants
- Les getters
- Les mutations d'état (modifications)
- Les commits
- Gestion des actions
- Différence avec Pinia

Exemples de travaux pratiques (à titre indicatif)

- Création d'un store de données d'état avec Vuex dans l'application en cours et mise à jour des données ; Exemple des mêmes fonctions avec Pinia et démonstration

Test avec Vue.js

- Pourquoi tester les applications Vue.js
- Les différents types de test
- Tester uniquement avec Vue.js
- Les frameworks à disposition (Jest...)

Exemples de travaux pratiques (à titre indicatif)

- Mise en oeuvre d'un test avec Jest

La performance

- Le code splitting
- Pourquoi le lazy loading est important dans Vue.js
- Le chargement à la demande
- Chargement lazy des routes avec "dynamic import"
- Les directives v-once et v-memo

Vuex et authentification

- Schéma de l'authentification
- Authentifier avec un Token
- JWT (JSON Web Token)
- Connexion / déconnexion

*Notre **référent handicap** se tient à votre disposition au 01.71.19.70.30 ou par mail à referent.handicap@edugroupe.com pour recueillir vos éventuels besoins d'aménagements, afin de vous offrir la meilleure expérience possible.*