

Angular avancé : maîtriser le framework

Maîtrisez les concepts et techniques avancés d'Angular pour concevoir des applications web robustes et évolutives.

Catégorie: Programmation · Durée: 3 jours

POUR QUI ?

- Développeurs front-end
- Développeurs de tout horizon
- Praticiens qui l'utilisent déjà au quotidien et veulent aller plus loin

Cette formation complète a pour objectif de faire passer vos compétences Angular au niveau supérieur. Dans la continuité de la formation Angular Fondamentaux, et en mettant l'accent sur les concepts et techniques avancés, vous apprendrez à concevoir des applications web robustes et évolutives avec Angular. Des patterns de conception avancés des composants au routage et à la gestion d'état, cette formation vous donnera les compétences nécessaires pour exceller dans le développement Angular. Les dernières fonctionnalités, telles que les signals et les composants standalone, sont incluses dans cette formation.

Prérequis

Connaissance préalable des fondamentaux d'Angular et expérience en développement web requises.

Programme

Rappels Angular

- Architecture d'une application
- Lazy loading au niveau des composants
- Automatiser la documentation
- Data binding manuel avec ChangeDetectorRef
- Analyser et optimiser une application existante

Mécanismes internes d'Angular

- ZoneJS : le concept
- Optimiser les cycles de rendu, exécuter du code hors de ZoneJS
- Pourquoi choisir RxJS
- Utiliser les observables
- Route guards et mécanismes de resolve
- CanActivate, CanActivateChild : protection des routes
- Pré-chargement avec resolve

Signals

- Introduction à l'API Signals : définition, lecture, écriture
- Signals en lecture seule
- Lien avec les observables
- Computed signals
- Effects
- Atelier pratique
- Réécrire un composant en utilisant des signals à la place des observables

Angular Standalone

- Pourquoi cette nouvelle architecture ?
- Mettre en place un composant standalone
- Une nouvelle architecture sans modules
- Migrer d'une architecture modulaire vers une architecture entièrement standalone

Internationalisation

- Marqueurs i18n
- Gérer les textes traduisibles
- Gestion de la pluralisation

Formulaires dynamiques : FormBuilder

- Créer des formulaires dynamiques avec ReactiveFormsModule
- FormControl, FormGroup, AbstractControl, FormArray
- Validation personnalisée et gestion des erreurs
- Créer des modèles de données
- Utiliser FormBuilder
- Générer dynamiquement des templates
- Abstraire la logique métier des formulaires dans des composants
- Implémenter la génération de formulaires et des cycles de validation avancés

Tests unitaires : bonnes pratiques et outils

- Jest vs Karma
- Tests d'intégration avec Cypress
- Couverture de code
- Behaviour Driven Development (BDD), Test Driven Development (TDD)
- Cas de test : pipes, composants, services, etc.
- TDD : développer une application en partant des tests unitaires

NgRx et gestion d'état

- Introduction à NgRx : store, actions, reducers, effects
- Architecture et flux de données unidirectionnel
- Selectors et entity adapters
- DevTools et débogage
- Modules d'état chargés en lazy loading
- Intégrer NgRx dans un module fonctionnel Angular

Progressive Web Apps (PWA)

- Qu'est-ce qu'une PWA ?
- Mettre en place le support PWA dans Angular
- Service workers et stratégies de cache
- Mode hors ligne et mécanisme de mise à jour
- Configuration du manifest
- Convertir l'application existante en PWA

Optimisation des performances et outils de débogage

- Utiliser Angular DevTools et le profilage du navigateur
- Déboguer la détection de changements avec ng.profiler
- trackBy et performance des directives structurelles
- Détection et correction des fuites mémoire
- Profiler les performances avant et après refactoring

Injection de dépendances avancée

- InjectionToken et @Inject
- Multi-providers et useClass/useFactory
- Portées de l'injection : root, platform, any
- Providers tree-shakable
- Contextes d'injection standalone
- Créer une configuration de services flexible et réutilisable

Sécurité et bonnes pratiques

- Protection XSS des templates et du DOM
- Jetons d'authentification dans un HttpInterceptor
- Route guards basés sur les rôles
- Interactions sécurisées avec les API et confiance envers le backend
- CSP et sanitizer Angular
- Sécuriser une application de bout en bout avec des guards et des interceptors