EduGroupe
Accompagner pour réussir

Angular 20 et versions antérieures - Fonctionnalités avancées

Référence : **OPS014C** Durée : **3 jours (21 heures)** Certification : **Aucune**

CONNAISSANCES PRÉALABLE

- Avoir suivi la formation OPS006D - Angular 20 et versions antérieures - Développement d'applications web ou posséder les connaissances et compétence équivalentes

PROFIL DES STAGIAIRES

- Développeurs Angular et chefs de projets

OBJECTIFS

- Définir le framework Angular, de manière approfondie

CERTIFICATION PRÉPARÉE

- Aucune

MÉTHODES PÉDAGOGIQUES

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

FORMATEUR

- Consultant-formateur expert Angular

MÉTHODES D'ÉVALUATION 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

ACCESSIBILITÉ DE LA FORMATION

- EduGroupe met en place un ensemble de dispositifs pour accueillir, accompagner et adapter ses formations aux personnes en situation de handicap (PSH).
- Notre référent(e) handicap se tient à votre disposition au 01.71.19.70.30 ou par mail à referent.handicap@edugroupe.com pour tout besoin d'aménagement, afin de vous offrir la meilleure expérience possible.

CONTENU DU COURS

1. Jour 1 - Matin

2. Rappel sur le fonctionnement d'Angular

- Vue d'ensemble de l'architecture actuelle : Préférence pour les composants autonomes ("standalone") ; Simplification du bootstrapping
- Présentation de l'état de l'écosystème : Usages actuels des modules ; Directives ; Pipes ; Services adaptés à Angular 20

3. Zone.js et Mode "Zoneless"

- Zone.js : explication de son rôle historique pour la détection de changements
- Mode zoneless désormais au coeur de l'architecture Angular 20 : Angular permet de s'affranchir de Zone.js pour un rendu plus rapide, moins de re-rendus inutiles et un contrôle manuel de la détection de changement grâce aux signals ou en déclenchant explicitement la détection ; Zoneless devient le mode recommandé (encore en "developer preview" mais déployable sur de vrais projets)

4. Nouveautés de la version 20

- Signal-based reactivity stable : Signaux réactifs ; Gestion manuelle et efficace du cycle de vie et des mises à jour d'UI sans Zone.js
- API de création dynamique de composants stabilisée createComponent(), directement compatible avec la nouvelle réactivité et les standalone components
- Syntaxe de template enrichie (opérateur **, template literals, opérateur in dans les expressions)

5. Rappel sur les "standalone components"

- Les "standalone components" sont pleinement matures depuis Angular 17+ : Exclusion de NgModule ; Simplification des dépendances ; Autonomie maximale

6. Exemple de travaux pratiques (à titre indicatif)

- *Création d'une application sur Angular 20, exploration des composants "standalone"*

7. Jour 1 - Après-midi

8. Gestion avancée de l'état avec NgRx

- NgRx 19 (compatible Angular 20) est la version actuelle, v20 est en préparation mais la v19 reste 100% compatible
- Pattern Flux revisité avec la nouvelle syntaxe "standalone" (API provideStore, provideEffects...)
- Stores, states, actions, effects : intégration simplifiée aux nouveaux composants autonomes et signaux
- Recommandation d'utiliser signals localement, et NgRx pour le state partagé / applications complexes

9. Exemple de travaux pratiques (à titre indicatif)

- *Développement d'un petit gestionnaire d'état avec NgRx, nouvelle intégration sans NgModule via les nouveaux providers*

10. Jour 2 - Matin

11. Routage avancé

- API de routage évoluée : Déclaration avec provideRouter ; Organisation modulaire facilitée
- Prise en charge étendue des secondary routes, routes relatives, options avancées (scroll smooth, options natives)
- Guards, resolvers accessibles et customisables
- Asynchronous redirects : fonction de redirection asynchrone (retourne une Promise ou un Observable)

12. Exemple de travaux pratiques (à titre indicatif)

- *Création d'une application multipage intégrant des guards et le chargement asynchrone*

13. Jour 2 - Après-midi

14. Angular Signals

- Signaux modifiables (writable signals) : Remplacement fluide de la gestion d'état locale au composant
- Signaux calculés (computed signals) : Réactivité fine ; Dépendance propre ; Exclu les abonnements RxJS inutiles
- Inputs signalés entre composants (communication parent / enfant efficace, sans "@Input" classique)
- Les signaux gèrent la notification de template de façon automatique, sans surconsommation de ressources

15. Exemple de travaux pratiques (à titre indicatif)

- *Conception d'une application manipulant signaux modifiables, calculés, et passages des signaux*

16. SSR (Server-Side Rendering)

- SSR activé par défaut sur projets nouveaux, prise en charge complète de l'hydratation incrémentale, event replay automatique pour une UX fluide même en SSR
- Configuration simplifiée du serveur pour le rendu dynamique et des pages statiques optimisées

17. Exemple de travaux pratiques (à titre indicatif)

- Mise en place d'une application SSR, gestion d'évènements réhydratés côté client

18. Jour 3 - Matin

19. Les Progressive Web Apps (PWA)

- Génération PWA via le schematic @angular/pwa, assets et manifest gérés automatiquement
- Mise à jour du service worker : Nouveaux outils de notification aux utilisateurs ; Stratégies de cache configurables ; Meilleure intégration avec Angular CLI

20. Exemple de travaux pratiques (à titre indicatif)

- *Création d'une PWA avec gestion de cache évoluée, détection de nouvelle version et stratégie hors-ligne*

21. Jour 3 - Après-midi

22. Internationalisation (i18n) et accessibilité

- Pipeline d'i18n maintenu et simplifié dans Angular 20 (extraction automatisée, support natif xlf, json...)
- Nouvelle gestion du pipe de régionalisation, meilleure gestion des formats (dates, monnaies...)
- Fonctionnement sur Ivy, comptabilité totale avec les composants standalone
- Renforcement de la prise en charge de l'accessibilité (composants natifs, directives, hooks d'accessibilité)

23. Exemple de travaux pratiques (à titre indicatif)

- *Internationalisation d'une application avec extraction, gestion de plusieurs langues, tests de comptabilité*