

Question 1 : la Structure d'un Projet Angular

1. Quel fichier sert de point d'entrée principal pour démarrer une application Angular ?

b) main.ts

2. Dans quel répertoire se trouvent généralement les composants, services et autres éléments de l'application Angular ?

c) src/app/

3. Quel est le rôle du fichier angular.json dans un projet Angular ?

b) Définir les configurations de construction et les options pour Angular CLI.

4. À quoi sert le fichier package.json dans un projet Angular ?

a) Il décrit les dépendances du projet et les scripts npm disponibles.

5. Quel est le rôle du fichier tsconfig.json dans un projet Angular ?

a) Configurer les options de compilation TypeScript pour le projet.

Question 2 : Création et l'Intégration de Composants Angular

1. Quelle commande Angular CLI permet de générer un nouveau composant nommé profil ?

c) ng generate component profil

2. Après avoir généré un composant, lequel des fichiers suivants n'est pas créé automatiquement ?

d) profil.service.ts

3. Quel décorateur est utilisé pour définir une classe en tant que composant Angular ?

b) @Component

4. Pour intégrer un nouveau composant dans l'application, que devez-vous ajouter dans le template du composant parent ?

a) La balise HTML personnalisée correspondant au sélecteur du composant.

5. Quelle est la commande pour créer un dossier de composants et générer un composant nommé utilisateur à l'intérieur en une seule commande ?

a) ng generate component composants/utilisateur

Question 3 : Liaison de Données Bidirectionnelle avec ngModel

1. Quelle est la syntaxe correcte pour utiliser ngModel dans un élément de formulaire pour la liaison de données bidirectionnelle ?

c) [(ngModel)]= "nom"

2. Pour que ngModel fonctionne dans un composant, quel module devez-vous importer dans app.module.ts ?

b) FormsModule

3. Quelle est la principale différence entre la liaison de données unidirectionnelle et bidirectionnelle ?

b) La liaison unidirectionnelle met à jour la vue lorsque le modèle change, tandis que la bidirectionnelle synchronise dans les deux sens.

4. Dans quel package Angular se trouve la directive ngModel ?

c) @angular/forms

5. Pourquoi est-il important d'utiliser la liaison de données bidirectionnelle dans les formulaires ?

b) Pour synchroniser automatiquement les données entre le modèle et la vue.

Question 4 : l'AppModule

1. Quel est le rôle principal de l'AppModule dans une application Angular ?

c) Il déclare les composants, directives, pipes et importe les modules nécessaires.

2. Quel décorateur est utilisé pour définir une classe en tant que module Angular ?

a) @NgModule

3. Dans quel tableau de l'@NgModule devez-vous ajouter le FormsModule pour utiliser ngModel ?

b) imports

4. Que se passe-t-il si vous oubliez d'importer un module nécessaire dans l'AppModule ?

c) Les fonctionnalités liées au module manquant ne fonctionneront pas, avec erreurs potentielles

5. Quel est le fichier qui contient la définition de l'AppModule par défaut dans un projet Angular ?

b) app.module.ts

Question 5 : QCM sur la Déclaration des Composants dans l'AppModule

1. Pourquoi est-il nécessaire de déclarer les composants dans l'AppModule ?

a) Pour que les composants puissent être utilisés dans les templates du module.

2. Dans quel tableau de l'@NgModule déclarez-vous les composants, directives et pipes de votre module ?

b) declarations

3. Que se passe-t-il si vous utilisez un composant dans un template sans l'avoir déclaré dans un module ?

b) Angular génère une erreur indiquant que le composant n'est pas une entité connue.

4. Est-il possible de déclarer un composant dans plusieurs modules simultanément ?

b) Non, un composant ne peut être déclaré que dans un seul module.

5. Quelle est la bonne pratique pour organiser les déclarations des composants dans les modules Angular ?

b) Créer des modules fonctionnels pour regrouper les composants liés.

Questions ouvertes pour évaluer les acquis :

1. Expliquez le rôle du fichier app.component.ts dans une application Angular et comment il s'intègre avec app.module.ts et main.ts.

Le fichier `app.component.ts` définit le composant racine de l'application Angular. C'est lui qui contient la logique de base, le template et le sélecteur `<app-root>` affiché dans `index.html`. Le fichier `main.ts` démarre l'application en lançant l'`AppModule`. L'`AppModule`, dans `app.module.ts`, déclare `AppComponent` et le place dans la section `bootstrap`, ce qui indique à Angular que ce composant doit être chargé en premier. Ainsi, `main.ts` lance l'application → `AppModule` configure l'application → `AppComponent` devient la première interface affichée.

2. Décrivez le processus complet de création d'un nouveau composant Angular et son intégration dans une application, en mentionnant les étapes clés et leur importance

Pour créer un composant, on utilise la commande Angular CLI `ng generate component nom`. Cela crée automatiquement quatre fichiers : `.ts`, `.html`, `.css` et `.spec.ts`. Le composant est ensuite ajouté dans la section `declarations` de l'`AppModule`, ce qui permet à Angular de le compiler et de le reconnaître. Pour l'utiliser, on insère sa balise (définie par le `selector` dans le fichier `.ts`) dans un template parent, comme `<app-nom></app-nom>`. À partir de là, le composant fait pleinement partie de l'interface de l'application.

3. Pourquoi est-il nécessaire d'importer le `FormsModule` pour utiliser `ngModel` ? Que contient ce module et comment contribue-t-il au fonctionnement des formulaires dans Angular ?

Pour utiliser `ngModel`, il faut importer `FormsModule`, car cette directive appartient au package `@angular/forms`. Sans ce module, Angular ne connaît pas `ngModel` et génère une erreur. `FormsModule` fournit toutes les fonctionnalités des formulaires template-driven : gestion des champs, synchronisation des valeurs, validations automatiques, et suivi des états (`touched`, `invalid`...). Il permet donc à Angular de relier correctement les données entre le composant TypeScript et le template HTML.

4. Expliquez ce qu'est la liaison de données bidirectionnelle et donnez un exemple concret de son utilisation dans un formulaire Angular.

La liaison de données bidirectionnelle synchronise automatiquement une valeur entre le template et le composant dans les deux sens. Quand l'utilisateur modifie un champ, la variable TypeScript change, et lorsqu'on modifie la variable dans le code, l'interface se met à jour aussi. Par exemple :

```
<input [(ngModel)]="nom">
```

5. Discutez des conséquences potentielles de ne pas déclarer un composant dans l'`AppModule` ou le module approprié. Comment cela affecte-t-il le comportement de l'application ?

Si un composant n'est pas déclaré dans l'AppModule (ou un autre module approprié), Angular ne le reconnaît pas. L'application affiche alors une erreur du type *"Component is not a known element"*. Le composant ne peut pas être rendu dans le DOM et la page peut cesser de fonctionner correctement. Déclarer le composant dans le module est donc indispensable pour que Angular puisse le compiler et l'utiliser dans l'application.