**Angular.json\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**2. projects**

Cette section contient la configuration spécifique au projet Angular nommé **pi**.

**pi**

* Nom du projet. Il est utilisé pour référencer ce projet (par exemple, ng build pi).

**projectType**

* Défini comme application, car c'est une application Angular (et non une bibliothèque).

**root et sourceRoot**

* **root** : Racine du projet. Ici, c'est la racine actuelle ("").
* **sourceRoot** : Emplacement des fichiers source du projet (src).

**prefix**

* Préfixe utilisé pour vos composants Angular. Par exemple, si vous générez un composant avec Angular CLI, le préfixe sera app- (ex. : <app-component>).

**3. architect**

Définit les tâches principales que Angular CLI peut effectuer pour ce projet.

**build**

Gère les options de construction de l'application Angular.

* **builder** : Définit le type de constructeur utilisé (@angular-devkit/build-angular:browser pour une application web).
* **options** :
  + **outputPath** : Dossier où les fichiers construits seront placés (dist/pi).
  + **index** : Fichier HTML principal (point d'entrée, ici src/index.html).
  + **main** : Fichier TypeScript principal (ici src/main.ts).
  + **polyfills** : Ajouts pour supporter des fonctionnalités manquantes dans certains navigateurs (zone.js est nécessaire pour Angular).
  + **tsConfig** : Fichier de configuration TypeScript pour l'application (tsconfig.app.json).
  + **assets** : Liste des fichiers ou dossiers copiés tels quels dans le dossier de sortie.
  + **styles** : Liste des fichiers CSS globaux (ici src/styles.css).
  + **scripts** : Scripts JavaScript externes ou personnalisés à inclure.

**Configurations**

* **production** :
  + Active des optimisations pour la production, comme le hachage des fichiers (outputHashing).
  + Définit des limites pour la taille des fichiers (ex. : maximum 500 Ko pour une alerte).
* **development** :
  + Désactive certaines optimisations pour faciliter le débogage, comme sourceMap (fichiers de mappage source pour le debug).

**defaultConfiguration**

* Définit la configuration par défaut pour la construction (ici, production).

 **outputPath** : Dossier où les fichiers construits seront placés (dist/pi).

 **index** : Fichier HTML principal (point d'entrée, ici src/index.html).

 **main** : Fichier TypeScript principal (ici src/main.ts).

 **polyfills** : Ajouts pour supporter des fonctionnalités manquantes dans certains navigateurs (zone.js est nécessaire pour Angular).

 **tsConfig** : Fichier de configuration TypeScript pour l'application (tsconfig.app.json).

 **assets** : Liste des fichiers ou dossiers copiés tels quels dans le dossier de sortie.

 **styles** : Liste des fichiers CSS globaux (ici src/styles.css).

*  **scripts** : Scripts JavaScript externes ou personnalisés à inclure.

**Résumé des Étapes pour Explorer**

1. **Comprendre les Entrées :**
   * Fichiers comme src/main.ts ou src/index.html sont les points de départ du projet.
2. **Configuration Personnalisée :**
   * Vérifiez les fichiers externes mentionnés (scripts, styles, assets) pour comprendre leur rôle.
3. **Modes de Construction :**
   * Testez les configurations avec :

**Résumé du rôle du fichier main.ts**

1. Charge la plateforme Angular pour les navigateurs (web).
2. Démarre l'application en chargeant le module racine (AppModule).
3. Fournit un mécanisme pour capturer et afficher les erreurs de démarrage.

**Lien avec le reste de l'application**

* **AppModule** (défini dans app.module.ts) contient :
  + La configuration des composants, services, et autres modules.
  + Le composant principal (AppComponent), qui est le point de départ visuel de l'application.
* Angular utilise main.ts pour initialiser l'application et connecter le fichier HTML principal (index.html) avec l'application Angular.

Le **fichier app.module.ts** définit le module principal de votre application Angular, le **AppModule**, et sert de base pour l'organisation des composants, des services et des dépendances nécessaires au bon fonctionnement de l'application.

1. Importation des modules Angular et tiers

 **NgModule** : Déclare un module Angular.

 **BrowserModule** : Nécessaire pour les applications Angular basées sur le navigateur.

 **ReactiveFormsModule** : Utilisé pour gérer les formulaires réactifs dans Angular.

 **HttpClientModule** : Permet la communication avec des API via HTTP.

 **BrowserAnimationsModule** : Nécessaire pour activer les animations Angular.

1. Importation des modules et composants spécifiques à l'application

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './components/dashboard/app.component';

* **AppRoutingModule** : Module de gestion des routes définissant la navigation entre les composants.
* **Composants importés** : Tous les composants de l'application sont importés individuellement. Cela inclut les widgets, les pages principales, et les fonctionnalités spécifiques (gestion, ajout, édition, etc.).

#### ****3. Déclaration des composants****

**@NgModule({**

**declarations: [**

**AppComponent,**

**SidebarComponent,**

**NavbarComponent,**

**FooterComponent,**

**PageHeaderComponent,**

**...**

**],**

* **declarations** : Liste de tous les composants, directives et pipes de l'application. Angular doit connaître chaque composant pour pouvoir les utiliser.

#### ****4. Importation des modules****

**imports** : Modules requis par l'application pour fonctionner. Cela inclut les modules Angular de base, les modules de formulaires, HTTP, animations, etc.

1. Fournisseurs de services

Démarrage de l'application

bootstrap: [AppComponent]

**bootstrap** : Déclare le composant racine qui sera chargé au démarrage de l'application. Ici, c'est AppComponent.