

# Angular

Chapitre 3: Structure et architecture d'un projet Angular

Enseignant: Mouhamed Amar

Licence 3

Université Amadou Hampaté BA de Dakar

Année 2022-2023



# Plan

1. Architecture globale
2. src
3. Fichiers de configurations du projet
4. Component (création, structure et utilisation)
5. Module (création, structure et exemple)

```
> .angular  
> .vscode  
> node_modules  
> src  
= .browserslistrc  
⚙ .editorconfig  
❖ .gitignore  
{ angular.json  
Ｋ karma.conf.js  
{ package-lock.json  
{ package.json  
ⓘ README.md  
{ tsconfig.app.json  
TS tsconfig.json  
{ tsconfig.spec.json
```

Configurations de Vscode

l'espace attribué afin de stocker des informations facilement accessibles dans le but d'accélérer certaines tâches

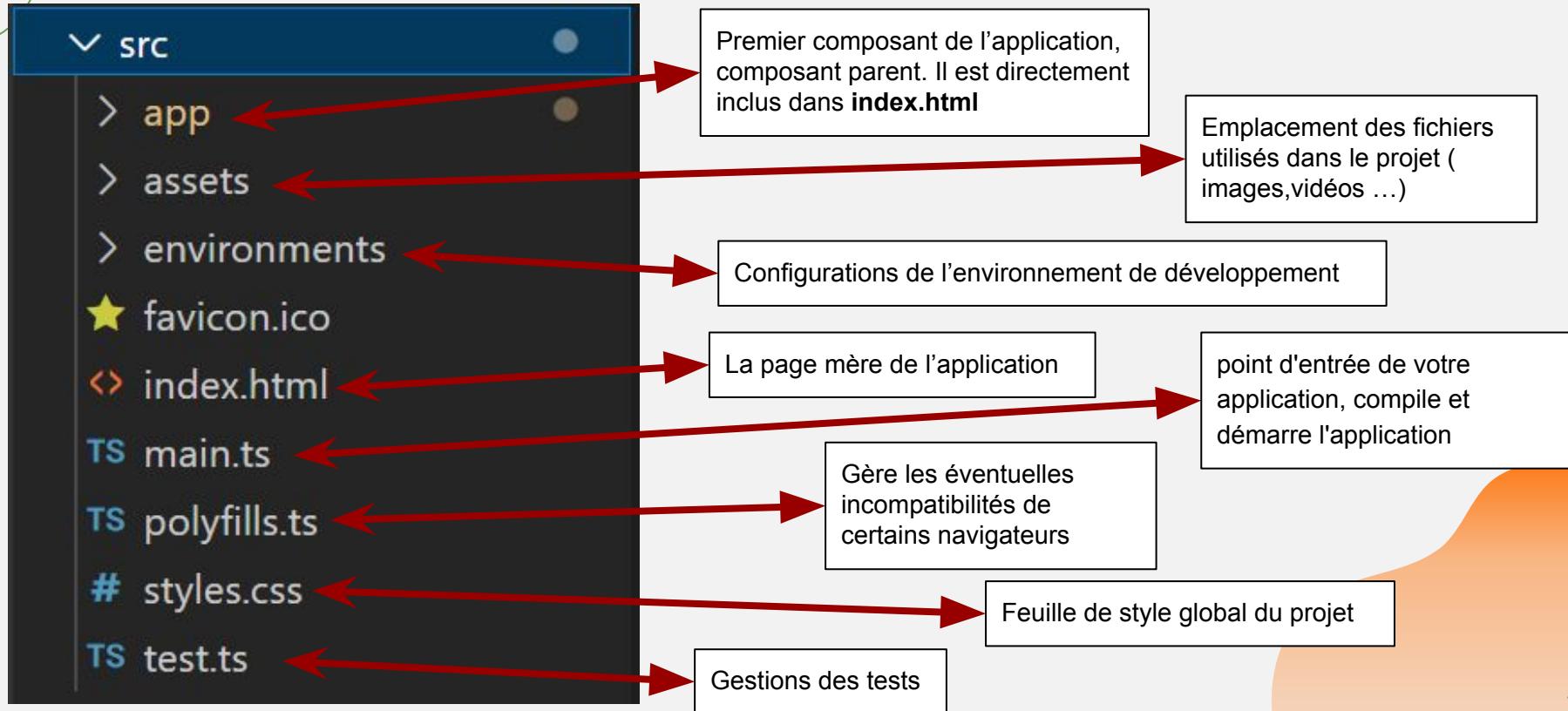
Emplacement des bibliothèques ou dépendances

Le corps de l'application Angular.

# Architecture globale

Configurations du projet Angular

# src ( corps du projet)



# Fichiers de configurations

≡ .browserslistrc

⚙ .editorconfig

❖ .gitignore

{ } angular.json

K karma.conf.js

{ } package-lock.json

{ } package.json

i README.md

{ } tsconfig.app.json

ts tsconfig.json

{ } tsconfig.spec.json

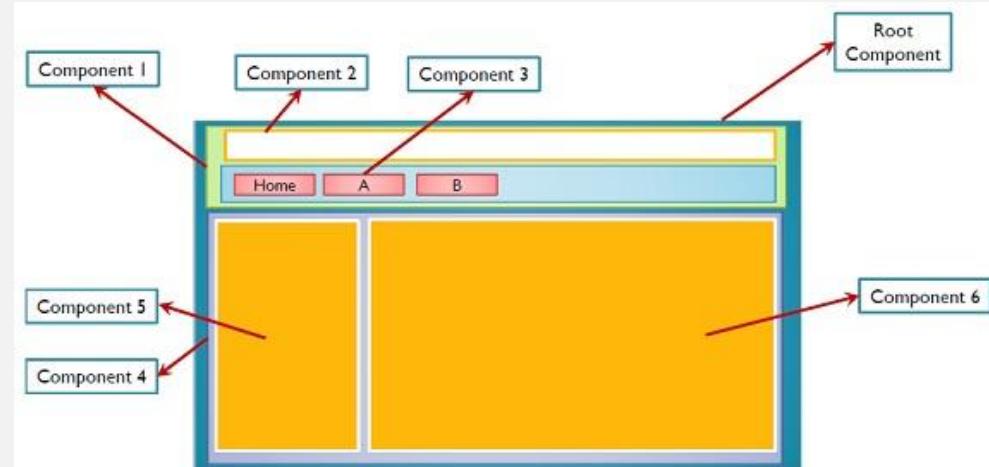
configurations par défaut de l'espace de travail utilisés par les outils de construction et de développement fournis par la Angular CLI

Ce fichier contient l'ensemble des dépendances du projet

spécifie les options de base du compilateur TypeScript et Angular

# Composant

Un composant est une **maison** dans la construction d'une application Angular supposée être le **quartier** ou la **ville** concerné(e). Toutes les zones d'une application Angular sont produites par des composants.



# Composant: création

ng generate component /apropos

ng g c /apropos

C:\WINDOWS\system32\cmd.exe

```
C:\xampp\htdocs\uahb\13_2022_2023\gestion_uahb\gestion_uahb_angular>ng g c apropos
CREATE src/app/apropos/apropos.component.html (22 bytes)
CREATE src/app/apropos/apropos.component.spec.ts (606 bytes)
CREATE src/app/apropos/apropos.component.ts (279 bytes)
CREATE src/app/apropos/apropos.component.css (0 bytes)
UPDATE src/app/app.module.ts (479 bytes)
```

# Composant: structure

▼ apropos

# apropos.component.css

<> apropos.component.html

TS apropos.component.spec.ts

TS apropos.component.ts

Code de mise en forme avec css

L'interface visible du composant

Les configurations pour les tests

La logique du composant avec **TypeScript**

# Composant: utilisation

Composant apropos  
Selecteur app-apropos

The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** GestionUahbAngular - Visual Studio Code
- File Explorer:** Shows the project structure:
  - app
  - > apropos (highlighted with a red box)
  - TS app-routing.module.ts
  - # app.component.css
  - <> app.component.html M
- Code Editor:** app.component.html (Preview tab)

```
<h1>Hello World</h1>
<app-apropos></app-apropos>
```
- Status Bar:** 1 <h1>Hello World</h1> 2 <app-apropos></app-apropos> 9

Two orange arrows point from the explanatory text above to the 'apropos' component in the file explorer and to the 'app-apropos' selector in the code editor.

Hello World

apropos works!

# Composant: utilisation

The screenshot shows a code editor interface with a dark theme. On the left is a sidebar with file navigation, and on the right is the main code editor area.

**File Tree (Sidebar):**

- src
  - app
  - assets
  - environments
  - favicon.ico
- index.html
- main.ts
- polyfills.ts
- styles.css
- test.ts
- .browserslistrc
- .editorconfig

**Code Editor (index.html):**

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>GestionUahbAngular</title>
    <base href="/">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" type="image/x-icon" href="favicon.ico">
  </head>
  <body>
    <app-root></app-root>
  </body>
</html>
```

**Annotations:**

- An orange arrow points from the text "Composant app" to the "app" folder in the file tree.
- An orange arrow points from the text "Selecteur app-root" to the opening tag "<app-root>" in the code editor.

**Text Overlay:**

Composant app  
Selecteur app-root

# Module

Un **module** est la **mairie** dans la construction d'une application Angular supposée être le **quartier** ou la **ville** concerné(e). C'est un mécanisme permettant de regrouper et d'organiser des **composants**, définir leurs dépendances, leurs visibilités ...

# Module création

ng generate **module** /enseignant --routing

ng g m /enseignant --routing

```
src
  app
    > apropos
  enseignant
    TS enseignant-routing.module.ts
    TS enseignant.module.ts
```

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [version 10.0.19044.2130]
(c) Microsoft Corporation. Tous droits réservés.

C:\xampp\htdocs\uahb\13_2022_2023\gestion_uahb\gestion_uahb-angular> ng g m /enseignant --routing
CREATE src/app/enseignant/enseignant-routing.module.ts (252 bytes)
CREATE src/app/enseignant/enseignant.module.ts (292 bytes)

C:\xampp\htdocs\uahb\13_2022_2023\gestion_uahb\gestion_uahb-angular>
```

Module **enseignant**

Module de routage **enseignant**

# Module: structure

ts enseignant.module.ts X

src > app > enseignant > ts enseignant.module.ts > EnseignantModule

```
1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3
4 import { EnseignantRoutingModule } from './enseignant-routing.module';
5
6
```

```
7 @NgModule({
8   declarations: [],
9   imports: [
10     CommonModule,
11     EnseignantRoutingModule
12   ]
13 })
14 export class EnseignantModule { }
```

Importations des dépendances

Décorateur qui montre qu'une classe est une **module** et fournit des métadonnées de configuration qui déterminent comment la module doit être traitée, instanciée et utilisée lors de l'exécution.

La classe proprement dite

# Module: exemple

```
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { AproposComponent } from './apropos/apropos.component';
```

Importations des dépendances

```
8 @NgModule({
9   declarations: [
10     AppComponent,
11     AproposComponent
12   ],
13   imports: [
14     BrowserModule,
15     AppRoutingModule
16   ],
17   providers: [],
18   bootstrap: [AppComponent]
19 })
```

Tous les **composants** qui dépendent directement de cette présente module

Importation des autres modules dont la présente **module** a besoins pour fonctionner

**Services** accessibles dans toutes les parties de l'application

```
20 export class AppModule { }
```

Importation des autres modules dont la présente **module** a besoins



# FIN

**De chapitre**

Pour plus de ressources, vous pouvez consulter:  
<https://angular.io>  
<https://blog.jant.tech>