

Workshop n°1 :

Mise en place de l'environnement

Objectifs

- Installer l'environnement de travail nécessaire pour Angular 18.
- Créer un projet Angular version 18
- Comprendre les différentes configurations possibles d'un projet Angular.

I- Étapes d'installation de l'environnement

- 1- Téléchargez et installez nodejs depuis le site : <https://nodejs.org/en/>.

Node.js Minimum : v18.13.0

Node.js recommandé : v20.x.x (LTS — Long Term Support)

- Pour vérifier que node est bien installé, tapez la commande **node -v** dans l'invite de commande.
- Pour avoir une idée sur les versions compatibles vous pouvez consulter le lien <https://angular.io/guide/versions>
- Le node package manager est installé automatiquement avec node. Vous pouvez vérifier la version installée de npm en tapant la commande **npm -v**

- 2- Installez l'outil CLI version 18 à l'aide de npm en tapant la commande suivante sur l'invite de commande **npm install -g @angular/cli@18**

RQ: Pour vérifier la version Angular installée, tapez la commande **ng version**

- 3- Installez un éditeur/IDE qui vous convient soit Visual Studio Code ou bien Web Storm

- 4- Créez un répertoire dans lequel vous enregistrez vos projets Angular, par exemple Angular_Workspace.

II- Crédit d'un projet Angular 100% standAlone

- 1- Ouvrez la fenêtre cmd dans le répertoire Angular_Workspace et tapez la commande : **ng new projectName**
- 2- Répondez par CSS à la première question et « Yes » ou « No » à la deuxième question.

```
C:\WINDOWS\system32>ng new mon-projet
? Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS ]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? (y/N) y
```

Figure 1 : Terminal Creation d'un projet Angular

- 3- Une fois la création est terminée, pour lancer votre projet, il faut accéder au projet crée via la commande **cd projectName** et puis taper la commande **ng serve** (ou bien la commande **ng serve -open**)

- 4- Une fois le projet est lancé, tapez l'url **localhost :4200**

Vous obtenez le résultat suivant :



Hello, myFirstProject

Congratulations! Your app is running. 🎉

[Explore the Docs](#)

[Learn with Tutorials](#)

[CLI Docs](#)

[Angular Language Service](#)

[Angular DevTools](#)



Figure 2 : page d'accueil d'un projet Angular

- 5- La structure du projet est la suivante:

The screenshot shows a file explorer on the left and a code editor on the right. The file explorer displays the following structure:

- projectCSR C:\Users\hp\Desktop\E
 - .idea
 - .vscode
 - node_modules library root (highlighted in yellow)
 - public
 - src
 - app
 - app.component.css
 - app.component.html
 - app.component.spec.ts (highlighted in green)
 - app.component.ts (highlighted in blue)
 - app.config.ts
 - app.routes.ts
 - index.html
 - main.ts
 - styles.css
 - .editorconfig
 - .gitignore
 - angular.json
 - package.json (highlighted in blue)
 - package-lock.json
 - README.md
 - tsconfig.app.json
 - tsconfig.json
 - tsconfig.spec.json

The code editor shows the content of app.component.ts:

```
1 import { Component } from '@angular/core';
2 import { RouterOutlet } from '@angular/router';
3
4 @Component({
5   selector: 'app-root',
6   standalone: true,
7   imports: [RouterOutlet],
8   templateUrl: './app.component.html',
9   styleUrls: ['./app.component.css'
10 })
11 export class AppComponent {
12   title : string = 'project';
13 }
```

Figure 3- La structure d'un projet Angular 100% standalone

III- Création d'un projet Angular Modulaire

- Ouvrez la fenêtre cmd dans le répertoire Angular_Workspace et tapez la commande : **ng new projectName --standalone=false**
- Répondez par CSS à la première question et « Yes » ou « No » à la deuxième question (voir Figure 1).
- Une fois la création est terminée, pour lancer votre projet, il faut accéder au projet créé via la commande **cd projectName** et puis taper la commande **ng serve** (ou bien la commande **ng serve -open**)

4. Une fois le projet est lancé, tapez l'url **localhost :4200** (voir figure 2).
5. La structure du projet est la suivante:

The screenshot shows a file explorer on the left and two code editors on the right. The file explorer displays a project structure with files like .idea, .vscode, node_modules, public, and src. The src folder contains app, index.html, main.ts, styles.css, and several ts files: app.component.css, app.component.html, app.component.spec.ts, app.component.ts, app.module.ts, and app-routing.module.ts. The code editor on the right has two tabs: 'app.module.ts' and 'app.component.ts'. A red arrow points from the 'app.module.ts' tab to the 'declarations' section of the code, which contains 'AppComponent'. Another red arrow points from the 'src' folder in the file explorer to the 'app.component.ts' code editor.

```

app.module.ts
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
7 @NgModule({
8   declarations: [
9     AppComponent
10   ],
11   imports: [
12     BrowserModule,
13     AppRoutingModule
14   ],
15   providers: [],
16   bootstrap: [AppComponent]
17 })
18 export class AppModule { }

```

```

app.component.ts
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-root',
5   templateUrl: './app.component.html',
6   styleUrls: ['./app.component.css'
7 })
8 export class AppComponent {
9   title = 'bigProject';
10 }

```

Figure 4- La structure d'un projet Angular Modulaire

IV- Standalone ou ngModule : Règle pratique pour choisir

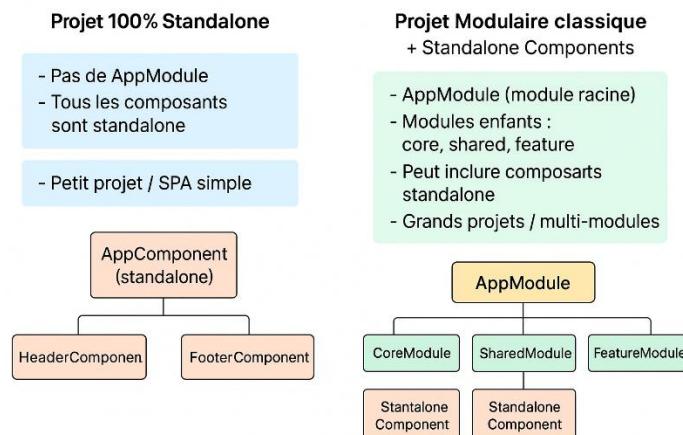


Figure 5 : les configurations possible d'un projet angular

Projet 100% Standalone	Projet modulaire classique (NgModule)
Caractéristique :	Caractéristique :
<ul style="list-style-type: none"> • Pas de AppModule du tout. • Tous les composants sont standalone. • Angular 14+ permet de créer des applications entièrement basées sur ces composants. 	<ul style="list-style-type: none"> • Basé sur AppModule et d'autres modules. • Peut inclure des composants standalone si besoin.
Quand l'utiliser :	Quand l'utiliser :
<p>Petits projets / prototypes / MVP : Rapidité de développement. Peu de modules à gérer.</p> <p>Applications SPA simples : Peu de dépendances et d'interactions complexes.</p> <p>Expérimentation avec la nouvelle architecture Angular : Tester les standalone components et voir les avantages du tree-shaking.</p>	<p>Grands projets avec plusieurs équipes. Facile de découper en modules : core, shared, feature.</p> <p>Applications complexes : Plusieurs routes, lazy loading, dépendances entre modules.</p> <p>Projets qui évoluent dans le temps : Le découpage en modules facilite la maintenance et l'extension.</p>
Avantages :	Mixte : possible d'utiliser des composants standalone à l'intérieur des modules pour profiter du tree-shaking et d'un lazy-loading granulaire.
<ul style="list-style-type: none"> • Plus léger et moderne. • Moins de fichiers NgModule à gérer. • Intégration plus facile avec des librairies modernes et du lazy-loading sur composants standalone. 	Avantages : <ul style="list-style-type: none"> • Architecture claire et maintenable. • Bien adapté pour les projets d'entreprise. • Compatible avec SSR, SSG et tous les outils Angular existants.
Limites :	Limites :
<ul style="list-style-type: none"> • Pour de très grands projets, la gestion de dépendances et de routing complexe peut devenir plus difficile. 	<ul style="list-style-type: none"> • Un peu plus lourd à mettre en place. • Plus de fichiers à gérer (NgModule).

Un résumé est donné dans le tableau suivant :

Type de projet	Approche recommandée
Petit projet / prototype rapide	100% standalone
Application SPA simple	Standalone ou modulaire légère
Grand projet / projet entreprise / multi-modules	Modulaire classique + possibilité de composants standalone
Besoin de SSR/SSG, routing complexe, lazy-loading	Modulaire + standalone components pour certaines parties

Résumé commandes :

Commande	Rôle	Quand l'utiliser
<code>node -v</code>	Vérifier la version de Node.js	Juste après l'installation de Node
<code>npm -v</code>	Vérifier la version de npm	Après l'installation de Node (npm vient avec Node)
<code>npm install -g @angular/cli@18</code>	Installer Angular CLI (v18) globalement	Une seule fois sur la machine (ou lors d'une mise à jour)
<code>ng version</code>	Vérifier les versions Angular/CLI/Node	Après l'installation de la CLI, et en cas de doute
<code>ng new < projectName ></code>	Créer un nouveau projet Angular 100% standalone	Au démarrage d'un projet
<code>ng new < projectName > --standalone=false</code>	Créer un nouveau projet Angular modulaire	Au démarrage d'un projet
<code>cd < projectName ></code>	Entrer dans le dossier du projet	Juste après la création du projet
<code>ng serve</code>	Lancer le serveur de dev	Pour démarrer l'app en local
<code>ng serve --open</code>	Lancer et ouvrir automatiquement le navigateur	Gain de temps au démarrage

Ressources sont disponibles ici: [lien](#)

<https://github.com/badi3a/AngularTraining/tree/workshop-01-setup>