




Tutorial 2 : Angular

Partie 1 : Préparation de l'environnement

Description des outils de développement requis

	<ul style="list-style-type: none"> Node.js est un environnement d'exécution JavaScript. Node.js nous permet d'utiliser le langage JavaScript sur le serveur. Il nous permet donc de faire du JavaScript en dehors du navigateur !
	<ul style="list-style-type: none"> Npm (node package manager) est un gestionnaire de dépendances. npm permet d'installer toutes les librairies, frameworks et outils dont a besoin un projet JavaScript Installé automatiquement avec NodeJs
 Command Line Interface	<ul style="list-style-type: none"> Angular CLI (Command Line Interface) : Un outil permettant de créer, construire, générer et tester vos applications et librairies Angular

Activité 1 : Préparation de l'environnement

1. Installer node.js (<https://nodejs.org/en>). Npm s'installe automatiquement avec nodejs
2. Pour installer Angular CLI, lancer une invite de commande cmd puis tapez la commande suivante puis tapez sur la touche «Entrer»: **npm install -g @angular/cli**

Si tout va bien, vous aurez un message comme ceci :

```
C:\Users\Hassene>npm install -g @angular/cli
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
C:\Users\Hassene\AppData\Roaming\npm\ng -> C:\Users\Hassene\AppData\Roaming\npm\node_modules\@angular\cli\bin\ng

> @angular/cli@10.1.7 postinstall C:\Users\Hassene\AppData\Roaming\npm\node_modules\@angular\cli
> node ./bin/postinstall/script.js

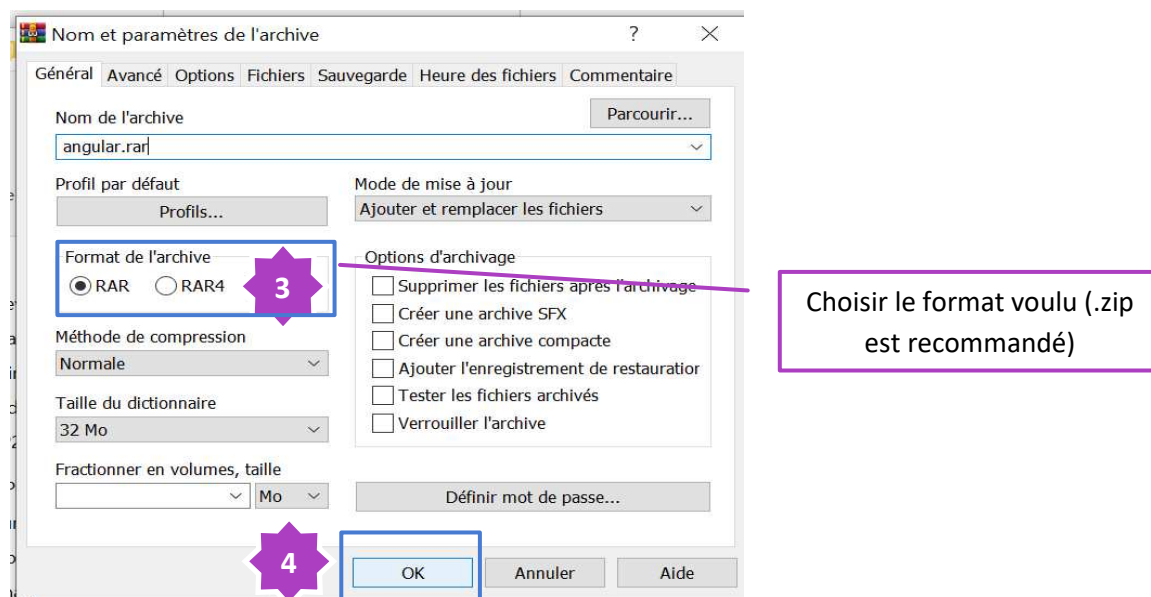
+ @angular/cli@10.1.7
added 3 packages from 2 contributors and updated 10 packages in 16.271s
```

3. Créer un nouveau projet angular nommé «**angular**» avec la commande (Accepter le paramétrage par défaut du projet :click sur le bouton Enter après chaque question proposée): **ng new angular**

4. Vérifier maintenant la création du dossier angular



Astuce : le dossier télécharger contient le squelette de base d'un projet Angular, essayer de la compresser (format .zip ou .rar) pour des futurs utilisations et ne pas être obligé de télécharger chaque fois vous êtes besoin de créer une nouvelle application Angular. Il suffit de l'extraire



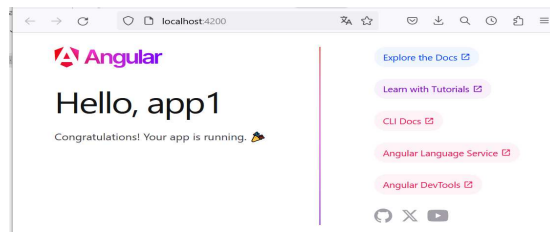
Activité 2 : Angular et Bootstrap

1. Ouvrir le dossier contenant le projet Angular à l'aide de Visual Studio Code
2. Pour lancer l'application, vous devez vous positionner dans le projet,
 - a. Pour lancer le serveur, exécuter la commande suivante dans le terminal (**CMD**) : `ng serve`
 - b. Pour lancer le serveur et l'application, exécuter : `ng serve -o`

NB : votre Single Page Application (SPA) est accessible à l'adresse **localhost :4200**

Remarque : clic Ctrl+C dans le terminal pour arrêter l'application

Si tout va bien, vous obtenez la figure suivante :



3. Ajouter bootstrap au projet :

- a. **Solution 1** : au fichier « index.html » à partir de son CDN (pour ne pas surcharger le projet) et pour qu'il soit reconnu par tous les autres composants inclus sous le composant principal.
- b. **Solution 2** : (<https://www.techiediaries.com/angular-bootstrap/>)
 - i. **Exécuter la commande : npm install bootstrap**
 - ii. Bootstrap sera installé dans le dossier node_modules/bootstrap. Vous devrez indiquer à Angular où les chercher. Par exemple, ajouter le référencement de bootstrap dans le fichier angular.json comme suit :

```
"styles": [  
    "src/styles.css",  
    "node_modules/bootstrap/dist/css/bootstrap.css"  
]
```

4. Le composant racine (**principal**) d'un projet est app.

NB : le fichier **index.html** charge le composant principal à partir de son sélecteur (**balise**) comme ainsi :

```
<body>  
  <app-root></app-root>  
</body>
```

5. Accéder au dossier **src/app** et modifier les fichiers **app.component.css** et **app.component.html** comme suit :

- a. Dans app.component.html :

```
<h1> Je suis le composant Principal App </h1>  
<button class="btn btn-success">Ok</button>
```

- b. Dans app.comonent.css : **h1 {color: blue}**

6. Sauvegarder les modifications effectuées puis vérifier de nouveau l'affichage

7. Créer la structure de page suivante :



- » Pour créer un nouveau composant, utiliser la commande suivante en spécifiant le nom du composant : **ng g c nom_composant --skip-tests**
- » Le sélecteur par défaut relatif à un Composant est de la forme : **app-nom_composant**.
- » **Pour utiliser un composant dans le fichier Html d'un autre composant, vous devez importer le composant en question dans le fichier .ts.**

Exemple, soit le composant <app-pub>. Dans le composant app, nous voulons afficher ce composant, voici le code nécessaire :

app.component.html	app.component.ts
<app-pub></app-pub>	<pre>import { PubComponent } from './pub/pub.component'; @Component({ .. imports: [...], PubComponent], templateUrl: './app.component.html', styleUrls: ['./app.component.css'] })</pre>