



NGLESSON

ALLEZ À L'ESSENTIEL POUR MAÎTRISER LE WEB

 [ACCEUIL](#)

 [INTÉGRATION WEB](#)

 [COURS](#)

 [SOLUTION TECHNIQUE](#)

 [LIVRES](#)

 [MÉDIA DIGITAL](#)

 [CITATIONS](#)

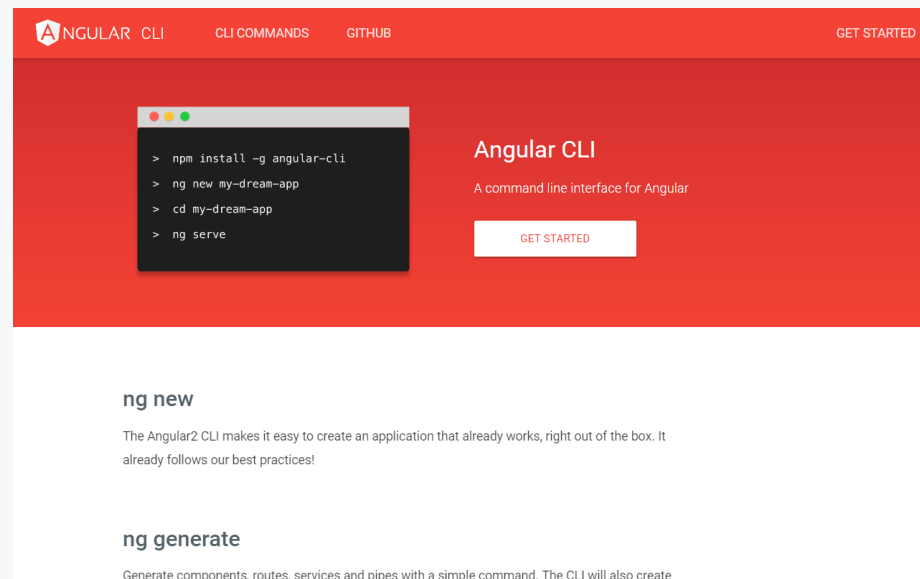
 [SERVICES EN LIGNE](#)

 [PLAN SITE WEB](#)

 [CONDITION GÉNÉRAL](#)

 / [Solutions techniques](#)

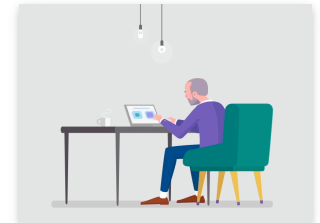
/ [Angular CLI : Une interface de ligne de commande pour Angular](#)



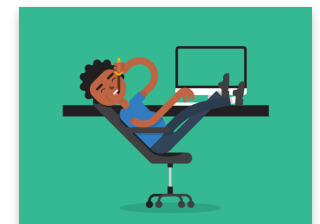
Angular CLI : Une interface de ligne de commande pour Angular




 **Développeur & Fondateur de la plateforme**



 **Devenir un développeur web en 3 mois**



 **la vie est plus simple quand on est**

Google nous propose un outil clé en main pour réaliser les tâches de développement les plus courantes. Ce projet est basé sur le projet Open Source ember CLI, qui a maintenant plus de trois ans. Grâce à notre Angular Cli, il est possible de :

- créer une application from scratch via un scaffolding ;
- Générer des squelettes des composants type Components... ;
- Builder un projet ;
- Lancer des tests de type « End-to-End » ou « unitaire » ;
- Proxyfier le back end ;
- Et beaucoup d'autres choses...

Angular-Cli est un outil en ligne de commande puissant qui va vous permettre de créer un projet contenant un ensemble plus que complet d'outils en vogue dans le monde JavaScript.

À la racine du projet, on retrouve un ensemble de fichiers de configuration :

- **package.json** : fichier déclarant les dépendances NPM tirées lors de l'installation du projet et nécessaire à la compilation et les tests.
- **.editorconfig** : ce fichier est issu du projet [EditorConfig](#). Il a pour but de maintenir une cohérence dans le code entre l'ensemble des éditeurs et IDE du marché. Le fichier fonctionne nativement sur certains éditeurs alors qu'un plugin sera nécessaire pour d'autres. Très peu d'éditeurs/IDE ne connaissent pas ce fichier ; c'est donc un standard de fait.

- **README.md** : fichier de présentation du projet au format [Markdown](#) utilisé notamment sur Github.
- **.gitignore** : fichier permettant de déclarer les fichiers qui ne doivent pas être commités sur le repository [Git](#).
- **karma.conf.js** : fichier de paramétrage du Test runner [Karma](#). Karma est un outil permettant de lancer des tests sur une série de browser/device automatiquement. Il est déjà configuré pour être lancé sur le navigateur Chrome avec le framework de test Jasmine.
- **protractor.conf.js** : fichier de paramétrage de l'outil de e2e [Protractor](#). E2E, ou end-to-end, est une discipline permettant de réaliser des tests d'intégration ; il est ainsi possible de réaliser des tests simulant un utilisateur final utilisant l'application dans un browser type Chrome.
- **tslint.json** : fichier définissant les règles de codage TypeScript. Tout comme le fichier .editorconfig, il est reconnu par la majorité des éditeurs de code.
- **angular-cli.json** : fichier de paramétrage central utilisé par Angular-cli. Ce fichier permet de définir où sont placés les sources de l'application, les différents fichiers de configuration, les scripts js et css tiers... Ce fichier est largement utilisé par la librairie webpack nouvellement ajoutée à Angular-Cli.
- **src** : à la racine du répertoire src, on retrouve les fichiers classiques index.html, favicon.ico, styles.css, mais également le main.ts (bootstrap d'Angular), le fichier de configuration de la compilation TypeScript tsconfig.json, un fichier de définition TypeScript typings.d.ts, et un ensemble de polyfills utiles à Angular,
- **src/app** : on retrouve les sources de notre premier projet, dont notre nouveau Component : AppComponent.
- **src/assets** : cet espace permet d'y placer tous les assets tels que les images. Lors de la compilation de l'application via Angular-cli, un dossier dist va être créé. Le contenu de ce dossier sera placé à la racine de dist.
- **src/environments** : les fichiers contenus dans ce dossier permettent de définir les variables spécifiques à chaque environnement d'exécution (prod, dev, integration).

Par défaut, l'environnement de dev sera utilisé (fichier environment.ts). Si l'on souhaite utiliser le fichier de production, il est nécessaire d'ajouter le paramètre -env=prod lors de l'appel de la commande ng build.

Installation

```
npm install -g @angular/cli
```

Usage

```
ng help
```

Génération et service d'un projet Angular via un serveur de développement

```
ng new PROJECT-NAME  
cd PROJECT-NAME  
ng serve
```

Accédez à <http://localhost:4200/>. L'application sera automatiquement rechargée

si vous modifiez l'un des fichiers source.

Vous pouvez configurer l'hôte et le port HTTP par défaut utilisés par le serveur de développement avec deux options de ligne de commande:

```
ng serve --host 0.0.0.0 --port 4201
```

Génération de composants, de directives, de tuyaux et de services

Vous pouvez utiliser la commande `ng generate` (ou simplement `ng g`) pour générer des composants angulaires:

```
ng generate component my-new-component
ng g component my-new-component # using the alias

# components support relative path generation
# if in the directory src/app/feature/ and you run
ng g component new-cmp
# your component will be generated in src/app/feature/new-cmp
# but if you were to run
ng g component ../newer-cmp
# your component will be generated in src/app/newer-cmp
# if in the directory src/app you can also run
ng g component feature/new-cmp
# and your component will be generated in src/app/feature/new-cmp
```

Vous pouvez trouver tous les plans possibles dans le tableau ci-dessous:

Scaffold	Usage
Component	<code>ng g component my-new-component</code>
Directive	<code>ng g directive my-new-directive</code>
Pipe	<code>ng g pipe my-new-pipe</code>
Service	<code>ng g service my-new-service</code>
Class	<code>ng g class my-new-class</code>
Guard	<code>ng g guard my-new-guard</code>
Interface	<code>ng g interface my-new-interface</code>
Enum	<code>ng g enum my-new-enum</code>
Module	<code>ng g module my-module</code>

angular-cli ajoutera automatiquement une référence aux composants, directives et

pipes dans app.module.ts. Si vous devez ajouter ces références à un autre module personnalisé, procédez comme suit:

- 1 - ng g module new-module pour créer un nouveau module
- 2 - appeler ng g composant nouveau-module / nouveau-composant

Cela devrait ajouter la nouvelle référence de composant, de directive ou de tuyau au nouveau module que vous avez créé.

Mise à jour de l'interface CLI angular

Si vous utilisez Angular CLI 1.0.0-beta.28 ou moins, vous devez désinstaller le paquet angular-cli. Cela devrait être fait en changeant le nom et la portée du paquet de angular-cli à @angular/cli:

```
npm uninstall -g angular-cli  
npm uninstall --save-dev angular-cli
```

To update Angular CLI to a new version, you must update both the global package and your project's local package.

Global package:

```
npm uninstall -g @angular/cli  
npm cache clean
```

```
# if npm version is > 5 then use `npm cache verify` to avoid errors (
npm install -g @angular/cli@latest
```

Local project package:

```
rm -rf node_modules dist # use rmdir /S/Q node_modules dist in Windows
npm install --save-dev @angular/cli@latest
npm install
```

Conseils de développement pour travailler sur CLI angular

```
git clone https://github.com/angular/angular-cli.git
cd angular-cli
npm link
```

```
ng new foo
cd foo
npm link @angular/cli
ng serve
```


Pour commencer par ici <https://cli.angular.io/>