Prerequisites

Both the CLI and generated project have dependencies that require Node 6.9.0 or higher, together with NPM 3 or higher.

Table of Contents

Installation

BEFORE YOU INSTALL: please read the prerequisites

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

Usage

```
ng help
```

Generating and serving an Angular project via a development server

```
ng new PROJECT-NAME

cd PROJECT-NAME

ng serve
```

Navigate to http://localhost:4200/. The app will automatically reload if you change any of the source files.

You can configure the default HTTP host and port used by the development server with two command-line options :

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

Generating Components, Directives, Pipes and Services

You can use the ng generate (or just ng g) command to generate Angular components:

```
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
```

You can find all possible blueprints in the table below:

Scaffold	Usage
Component	ng g component my-new-component
Directive	ng g directive my-new-directive
Pipe	ng g pipe my-new-pipe
Service	ng g service my-new-service

Scaffold	Usage
Class	ng g class my-new-class
Guard	ng g guard my-new-guard
Interface	ng g interface my-new-interface
Enum	ng g enum my-new-enum
Module	ng g module my-module

angular-cli will add reference to components, directives and pipes automatically in the app.module.ts. If you need to add this references to another custom module, follow this steps:

1. ng g module new-module to create a new module
2. call ng g component new-module/new-component

This should add the new component, directive or pipe reference to the new-module you've created.

Updating Angular CLI

If you're using Angular CLI 1.0.0-beta.28 or less, you need to uninstall angular-cli package. It should be done due to changing of package's name and scope from angular-cli to @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 (or to avoid using --force)

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

Local project package:

```
rm -rf node_modules dist # use rmdir /S/Q node_modules dist in Windows Command Prompt; use rm -r -fo node_modules,dist in Windows PowerShell

npm install --save-dev @angular/cli@latest

npm install
```

If you are updating to 1.0 from a beta or RC version, check out our 1.0 Update Guide.

You can find more details about changes between versions in the Releases tab on GitHub.

Development Hints for working on Angular CLI

Working with master

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

cd angular-cli

npm link
```

npm link is very similar to npm install -g except that instead of downloading the package from the repo, the just cloned angular-cli/ folder becomes the global package. Additionally, this repository publishes several packages and we use special logic to load all of them on development setups.

Any changes to the files in the angular-cli/ folder will immediately affect the global @angular/cli package, allowing you to quickly test any changes you make to the cli project.

Now you can use @angular/cli via the command line:

```
ng new foo

cd foo

npm link @angular/cli

ng serve
```

npm link @angular/cli is needed because by default the globally installed @angular/cli just loads the local @angular/clifrom the project which was fetched remotely from npm. npm link @angular/cli symlinks the global @angular/cli package to the local @angular/cli package. Now the angular-cli you cloned before is in three places: The folder you cloned it into, npm's folder where it stores global packages and the Angular CLI project you just created.

You can also use ng new foo --link-cli to automatically link the @angular/cli package.

Please read the official npm-link documentation and the npm-link cheatsheet for more information.

To run the Angular CLI test suite use the node tests/run_e2e.js command. It can also receive a filename to only run that test (e.g. node tests/run_e2e.js tests/e2e/tests/build/dev-build.ts).

As part of the test procedure, all packages will be built and linked. You will need to re-run npm link to re-link the development Angular CLI environment after tests finish.