

Setup

<code>npm i -g @angular/cli</code>	⚡ Install latest version
<code>ng -v</code>	📄 check your CLI version
<code>ng --help</code>	❓ show all commands in terminal
<code>ng doc (component)</code>	📖 open Angular documentation

Main Commands

<code>ng new</code>	<code>ng new myApp</code>	create a new project
<code>ng serve</code>	<code>ng s -o</code>	compile and open
<code>ng generate</code>	<code>ng g</code>	create new component/route/service
<code>ng build</code>	<code>ng b</code>	build for deployment in dist folder
<code>ng test</code>		test spec files
<code>ng e2e</code>	<code>ng e</code>	end-to-end tests
<code>ng lint</code>	<code>ng l</code>	run linting
<code>ng update</code>		identify dependencies to update
<code>ng add</code>		add libraries like Material

Common Flags

COMMAND	FLAG	ALIAS	DESCRIPTION
All	<code>--help</code>		list flags (new/s/g)
All	<code>--dry-run</code>	<code>-d</code>	show potential file changes
New	<code>--skip-tests</code>	<code>-S</code>	no spec files
New	<code>--skip-install</code>		don't install dependencies
New	<code>--prefix</code>	<code>-p ng</code>	prefix for all selectors
New	<code>--skip-git</code>	<code>-g</code>	don't add git
New	<code>--style scss</code>		set up using SASS
New	<code>--routing</code>		generates routing module
Serve	<code>-prod</code>		run from production
Serve	<code>-open</code>	<code>-o</code>	opens in browser

Notes

Certain *Blueprints (components/modules)* create a **folder** - unless a folder has already been created. Use the `--flat` flag to prevent a folder from being created. **Service & Interface** do not create folders.

Remember to consider **where components should reside**. You may want to add a login component inside an account module. This would be done by first creating the module `ng g m account --routing` then adding the login component inside account and importing it the correct module `ng g c account/login -m=account`.

Yarn vs NPM

Enable Yarn	<code>ng config -g cli.packageManager yarn</code>
Enable NPM	<code>ng config -g cli.packageManager npm</code>

ng update

<code>ng update @angular/core</code>	update core/rxJS/TypeScript
<code>ng update @angular/material</code>	update Material

ng add - Schematics

<code>ng add @angular/pwa</code>	app manifest + service worker
<code>ng add @ng-bootstrap/schematics</code>	add ng-Bootstrap
<code>ng add @angular/material</code>	install Material
<code>ng add @clr/angular@next</code>	install Clarity Design

Material Schematics

<code>ng g @angular/material:material-nav -name=nav</code>	Nav Menu
<code>ng g @angular/material:material-dashboard --name=dash</code>	Cards
<code>ng g @angular/material:material-table --name=table</code>	Table

Makes it easy to add and update 3rd party libraries.



Blueprints

component	c
service	s
pipe	p
interface	i
module	m
class	class
--flat	don't create a folder
--spec false	-S no spec files
--inline-style	-s no CSS files
--inline-template	-t no HTML files
--skip-import	don't add to module
--dry-run	-d report files, don't write
-m	-m specify which module to import into

Blueprints - The concept angular uses to generate code.

Blueprint Examples

<code>ng g m account</code>	create account Module along w/ folder
<code>ng g c account/login -m account</code>	new login Component added to account module
<code>ng g s account/login -m account</code>	new login Service added to account module
<code>ng g i models/person</code>	new person Interface added to models
<code>ng new myApp -S -g --routing</code>	new project no spec or git w/ routing



By **jakblak**
cheatography.com/jakblak/

Not published yet.
Last updated 13th July, 2018.
Page 2 of 2.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>