ngX Starter Kit

Web project starter kit including modern tools and workflow based on angular-cli, best practices from the community, a scalable base template and a good learning base.

Generated using ngX-Rocket.

Benefits

- Quickstart a project in seconds and focus on features, not on frameworks or tools
- Industrial-grade tools, ready for usage in a continuous integration environment and DevOps
- Scalable architecture with base app template including example components, services and tests

Getting started

1. Go to project folder and install dependencies:

```
npm install
```

2. Launch development server, and open `localhost:4200` in your browser:

```
npm start
```

Project structure

```
dist/                        compiled version
docs/                        project docs and coding guides
e2e/                         end-to-end tests
src/                         project source code
|- app/                      app components
|  |- @core/                 core module (singleton services and single-use
components)
|  |- @shared/               shared module  (common components, directives and
pipes)
|  |- app.component.*        app root component (shell)
|  |- app.module.ts          app root module definition
|  |- app-routing.module.ts  app routes
|  +- ...                    additional modules and components
|- assets/                   app assets (images, fonts, sounds...)
|- environments/             values for various build environments
|- theme/                    app global scss variables and theme
```

```
  |- translations/          translations files
  |- index.html             html entry point
  |- main.scss              global style entry point
  |- main.ts                app entry point
  |- polyfills.ts           polyfills needed by Angular
  +- test.ts                unit tests entry point
  reports/                  test and coverage reports
  proxy.conf.js             backend proxy configuration
```

Main tasks

Task automation is based on NPM scripts.

Tasks	Description
npm start	Run development server on `http://localhost:4200/`
npm run build [-- --env=prod]	Lint code and build app for production in `dist/` folder
npm test	Run unit tests via Karma in watch mode
npm run test:ci	Lint code and run unit tests once for continuous integration
npm run e2e	Run e2e tests using Protractor
npm run lint	Lint code
npm run translations:extract	Extract strings from code and templates to `src/app/translations/template.json`
npm run docs	Display project documentation

When building the application, you can specify the target environment using the additional flag `--env <name>` (do not forget to prepend `--` to pass arguments to npm scripts).

The default build environment is `prod`.

Development server

Run `npm start` for a dev server. Navigate to `http://localhost:4200/`. The app will automatically reload if you change any of the source files. You should not use `ng serve` directly, as it does not use the backend proxy configuration by default.

Code scaffolding

Run `npm run generate -- component <name>` to generate a new component. You can also use `npm run generate -- directive|pipe|service|class|module`.

If you have installed angular-cli globally with `npm install -g @angular/cli`, you can also use the command `ng generate` directly.

Additional tools

Tasks are mostly based on the `angular-cli` tool. Use `ng help` to get more help or go check out the Angular-CLI README.

What's in the box

The app template is based on HTML5, TypeScript and Sass. The translation files use the common JSON format.

Tools

Development, build and quality processes are based on angular-cli and NPM scripts, which includes:

- Optimized build and bundling process with Webpack
- Development server with backend proxy and live reload
- Cross-browser CSS with autoprefixer and browserslist
- Asset revisioning for better cache management
- Unit tests using Jasmine and Karma
- End-to-end tests using Protractor
- Static code analysis: TSLint, Codelyzer, Stylelint and HTMLHint
- Local knowledgebase server using Hads
- Automatic code formatting with Prettier

Libraries

- Angular
- Bootstrap 4
- Font Awesome
- RxJS
- ng-bootstrap
- ngx-translate
- Lodash

Coding guides

- Angular
- TypeScript
- Sass
- HTML
- Unit tests
- End-to-end tests

Other documentation

- I18n guide
- Working behind a corporate proxy
- Updating dependencies and tools

- Using a backend proxy for development
- Browser routing

License

MIT