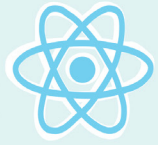


🐦 @nxdevtools
📌 nx.dev

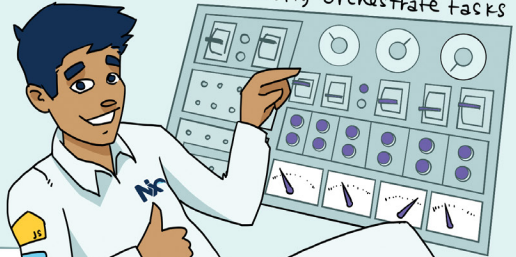
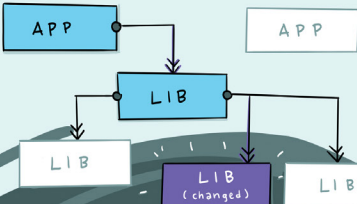


React

it's easy to
get started!

At its core, Nx helps you to...
build and verify only the code
that's affected by your changes

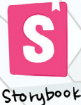
and easily orchestrate tasks



...but it's also
**POWERFULLY
EXTENSIBLE**

add only the plugins you
need, for a rich developer
experience

NEXT.JS



Storybook



React
Native



ESLint



Cypress



Jest

With
**BUILT-IN
LOCAL
CACHING**

Nx stores the results of your targets' computations (terminal output & artifacts) and replays them - speeding up code verification & builds!

Nx is growing quickly,
with over

24 MILLION
NPM downloads
/ month

come join us!
- your friends



drawn by @hixcodes

WARP SPEED AHEAD! 🚀

A quick guide to Nx + React

Getting Started

Create a new Nx workspace to host your React application.

```
npx create-nx-workspace <workspace-name> --preset=react
```

Install Nx globally or use npx

You can install Nx globally with: `npm i nx -g`

Or just prefix your commands with `npx`.



NX CONSOLE IDE INTEGRATION

Use the dedicated VSCode and JetBrains extensions to explore and run generators and navigate the workspace much more easily.

<https://nx.dev/features/integrate-with-editors>

Add Nx to an existing repo

If you already have a npm/yarn/pnpm workspace, you can easily add Nx on top for task scheduling and advanced caching while you preserve full control over your scripts.

```
npx nx@latest init
```

Read more: <https://nx.dev/recipes/adopting-nx/adding-to-monorepo>

Migrate from CRA

Easily migrate from a create-react-app setup to Nx. `npx nx@latest init`

Read more: <https://nx.dev/recipes/adopting-nx/migration-cra>

Run tasks with Nx

```
nx <target> <project-name>
```

Example: Run tests for my-project: `nx test my-project`

Run only affected Projects

Nx allows only running targets like test, lint, build on projects that have been changed in a given PR, using so-called “affected commands”.

```
nx affected --target=<target>
```

Example: run all tests for the projects that have been changed in the current PR:

```
nx affected --target=test
```

Setup Remote Distributed Caching

All Nx workspaces come with local caching abilities, but the real benefit comes when this cache is distributed and shared with other team members and your CI servers. This is what Nx Cloud does. Here's how to connect your Nx workspace:

```
nx connect-to-nx-cloud
```

<https://nx.dev/features/share-your-cache>



POWER UP YOUR NX DEVELOPMENT WITH NX CLOUD

Nx Cloud helps save computation time and speeds up CI times by distributing Nx's cache and task execution across multiple machines.

Visit nx.dev/pricing for pricing and additional info.

Setup Distributed Task Execution on CI

Generate a complete CI configuration that uses Nx Cloud's distributed task execution to optimally parallelize tasks focusing on maximum agent utilization!

```
nx generate @nx/workspace:ci-workflow --ci=github
```

<https://nx.dev/features/distribute-task-execution>

@nx/react plugin

Generators

```
nx generate <package-name>:<generator> <options>
```

Tip: use `--dry-run` to simulate the execution of the generator.

Generate a new Application

```
nx g @nx/react:application <name>
```

Generate a new Library

```
nx g @nx/react:library <name>
```



APPS AND LIBS

Nx recommends having thin applications and suggests moving features into dedicated workspace libraries. This helps with establishing a cleaner, more defined architecture as well as speeding up CI times.

Generate a component

```
nx g @nx/react:component hello-world --project=mylib
```



GENERATORS AT YOUR FINGERTIPS

With Nx Console installed, you can right-click on a project in Visual Studio Code and directly invoke the 'Nx Generate' menu item.

Generate a Module Federation Setup

Looking into Microfrontends or just want to speed up your React builds? Generate a new Webpack Module Federation app like

```
nx g @nx/react:host shell --remotes=featureA,featureB,featureC
```

<https://nx.dev/recipes/module-federation/faster-builds>

Powerful Integrations: Cypress, Storybook, Tailwind

All apps come with a **Cypress** setup out of the box. You can run Cypress based e2e tests with:

```
nx e2e <project-name>--e2e
```

Add **Storybook** support with a single command, including generating stories for components and setting up Cypress based e2e tests for them.

```
nx g @nx/react:storybook-configuration <project-name>
```

Add **Tailwind** support by running:

```
nx g @nx/react:setup-tailwind <project-name>
```

Automated updates

```
nx migrate latest
```



LEARN MORE!

 **Nx website:** <https://nx.dev>

 **Nx on Github:** <https://github.com/nrwl/nx>

 **Nx Community Discord:** <https://go.nx.dev/community>

 **Nx YouTube Channel:** <https://www.youtube.com/@nxdevtools>

 **Nx Twitter:** <https://twitter.com/nxdevtools>