React Native Beginner’s Guide: CLI vs Expo & Setup

Subtitle: Learn how to set up your React Native environment and choose the right workflow for building mobile apps.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app

Introduction

React Native allows you to build mobile apps for **iOS and Android** using **JavaScript and React**. Beginners often struggle to choose between **React Native CLI** and **Expo**, and how to set up their environment correctly. This guide simplifies that process and gets you coding fast.

Step 1: Understanding React Native Workflows

React Native offers **two main ways** to start a project:

1. React Native CLI

- Gives **full control** over the project.
- Requires **manual setup** for iOS and Android environments.
- Best for **complex apps with native modules** or custom functionality.
- Pros: Flexibility, full native access.
- Cons: Longer setup, more configuration.

2. Expo

- **Managed workflow**: handles native dependencies for you.
- Easy to start with **one command** and instant live preview.
- Best for **beginners and simple-to-medium apps**.
- Pros: Quick setup, hot reload, over-the-air updates.
- Cons: Limited access to custom native modules without ejecting.

Example Decision:

- Building a simple notes app → Use **Expo**.

- Building a social media app with camera & push notifications → Use **CLI**.

Step 2: Environment Setup

1. Node.js & NPM/Yarn

- Install **Node.js**: https://nodejs.org
- Install **Yarn (optional but recommended)**: `npm install -g yarn`

2. React Native CLI Setup

1. Install CLI globally:

```
npm install -g react-native-cli
```

2. Install **Android Studio** for Android emulator.
3. For iOS (Mac only), install **Xcode**.
4. Create a project:

```
npx react-native init MyApp
```

5. Run app:

```
npx react-native run-android
npx react-native run-ios
```

3. Expo Setup

1. Install Expo CLI globally:

```
npm install -g expo-cli
```

2. Create a new project:

```
expo init MyApp
```

3. Start project:

```
cd MyApp
```

```
cd MyApp
expo start
```

4. Scan the QR code with **Expo Go app** on your phone.

Step 3: Project Structure Overview

React Native projects typically include:

- `App.js` – Main entry file.
- `package.json` – Dependencies & scripts.
- `node_modules/` – Installed packages.
- `android/` & `ios/` – Native platform folders (CLI only).
- `assets/` – Images, fonts, icons.

Exercise: Open your first project and locate `App.js`. Modify the default text to display your name.

Step 4: Core Concepts for Beginners

- **Components:** Reusable UI pieces (`View`, `Text`, `Button`).
- **State & Props:** Store and pass dynamic data.
- **Styling:** Use `StyleSheet` or inline styles (no CSS).
- **Navigation:** React Navigation library for screens.
- **APIs & Fetch:** Communicate with backend services.

Mini Exercise: Create a screen with:

- Header text: “Hello React Native”
- A button that logs “Button pressed” in the console.

Step 5: Tips for Smooth Development

- Use **Expo** if you want fast iteration and beginner-friendly setup.
- Use **CLI** when you need **custom native code** or **performance optimization**.
- Always keep **Node.js, React Native, and Expo CLI updated**.
- Test on **both iOS and Android** for consistency.

- Test on **both iOS and Android** for consistency.

Key Takeaways

- React Native can target both **iOS and Android** using a single codebase.
- **CLI = more control**, **Expo = faster start**.
- Proper **environment setup** is essential to avoid headaches.
- Start small, explore components, and gradually integrate native features.

Visit **haas.dev** for more detailed React Native tutorials, tips, and full project guides.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app
