

Lab: Setting up the Environment using Expo Snack



Estimated time needed: 20 minutes

What you will learn

This app is a simple demonstration of how to set up and run a React Native project using Expo Go and Expo Snack. It allows users to easily test and develop mobile apps in real time. You will gain hands-on experience in setting up a React Native development environment using Expo Go and Expo Snack. You will also understand how to run and preview projects on a mobile device without complex configurations.

Learning objectives

After completing this lab, you will be able to:

- Install and configure Expo Go on a mobile device
- Use Expo Snack to write and test React Native code in the browser
- Learn how to preview and update mobile apps in real time with Expo Go

Expo Snack

Expo Snack is an online code editor and development tool that lets you write, preview, and share React Native apps directly in your web browser. It allows you to test your app on virtual and real devices without installing anything locally, providing a quick and simple way to experiment with React Native. It reduces the need for the Android Emulator, which tests React Native applications for Android devices. Also, it reduces the need for the iOS Simulator, which tests React Native applications for Apple iOS devices.

Account setup on Expo Dev

1. First, you need to have an account on <https://expo.dev/> if you have not signed up yet. Otherwise, log in to the account.

A screenshot of a web browser showing the Expo Dev dashboard. The address bar shows 'https://expo.dev'. The dashboard features a navigation bar with icons for back, forward, refresh, and a search bar labeled 'Search'. To the right of the search bar are keyboard shortcut keys 'Ctrl' and 'K'. Further right are links for 'Docs', 'Tools', and 'EAS'. The main content area of the dashboard is currently blank.

2. After login, your dashboard for Expo Dev will open as in the screenshot below.

The screenshot shows the Expo Dashboard interface. On the left, there is a sidebar with the following navigation items:

- Dashboard (selected)
- Projects
- Snacks
- Usage
- Account settings (expanded)
 - Overview
 - Members
 - Access tokens
 - Credentials
 - Apple devices

The main content area is titled "Dashboard" and contains the following sections:

- Projects**: A folder icon followed by the text "Projects". To the right is a link "All Projects →". Below this, it says "No projects" and "Create a project to get started with Expo". A "Create a Project" button is present.
- Recent activity**: A section with a small circular icon containing a checkmark and the text "Recent activity".
- A small notification icon in the top right corner indicates "1 notifications".

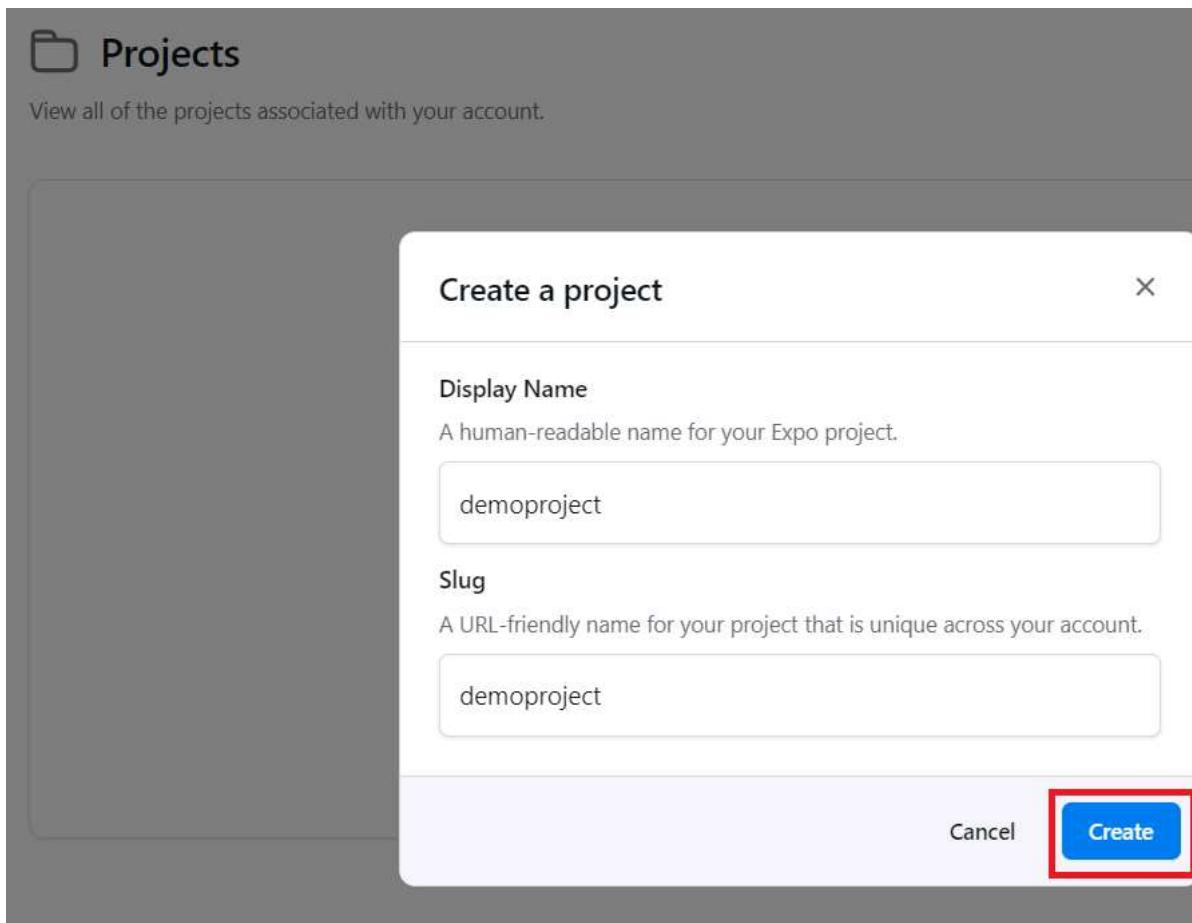
3. The **Projects** section will be empty. To create a new project, click **Create a Project**.

The screenshot shows the Expo Dashboard interface. On the left, there is a sidebar with the following menu items:

- Search bar with 'Search' placeholder and keyboard shortcuts 'Ctrl + K'.
- Account section showing a profile picture of 'richaarora9988'.
- Dashboard (selected)
- Projects
- Snacks
- Usage
- Account settings (dropdown menu)
 - Overview
 - Members
 - Access tokens
 - Credentials
 - Apple devices

The main dashboard area has a title 'Dashboard' with a list icon. It displays the 'Projects' section, which shows a folder icon and the text 'No projects'. Below it, a call-to-action says 'Create a project to get started with Expo'. A prominent 'Create a Project' button is located below this text, enclosed in a red rectangular box. To the right of the main dashboard, there is a 'Recent activity' section with a small icon and the text 'Make some builds, update your apps...'. The overall background is white with light gray borders between sections.

4. Then, it will display a pop-up dialog box where you need to provide the project name. Then click **Create**.



5. After creating a new project, one more pop-up dialog box will appear, as in the screenshot below.



Successfully created project

Start developing your project

You just created a new EAS project. Now, it's time to link this to your local project. There are two ways to accomplish this:

Create a new project

```
Terminal Copy
- npm install --global eas-cli
- npx create-expo-app demoproject
- cd demoproject
- eas init --id c5fa9ec3-f901-4add-b6b7-352ddbf3cc6f
```

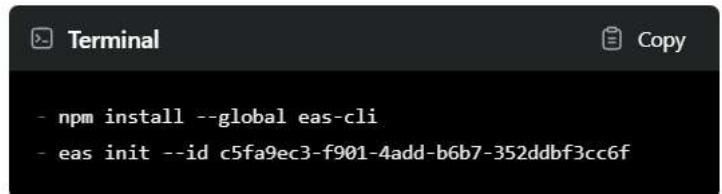
Link an existing codebase

[View project →](#)

6. The pop-up box in step 5 will show you two ways to include this newly built project with your project in localhost. The first step will guide you in creating a new project in your localhost code editor's terminal. To do this, you need to follow the commands step-by-step.
7. Another useful approach is to use the project already running on your localhost. You can connect your project in Expo Dev with the one already hosted on your localhost.

Link an existing codebase

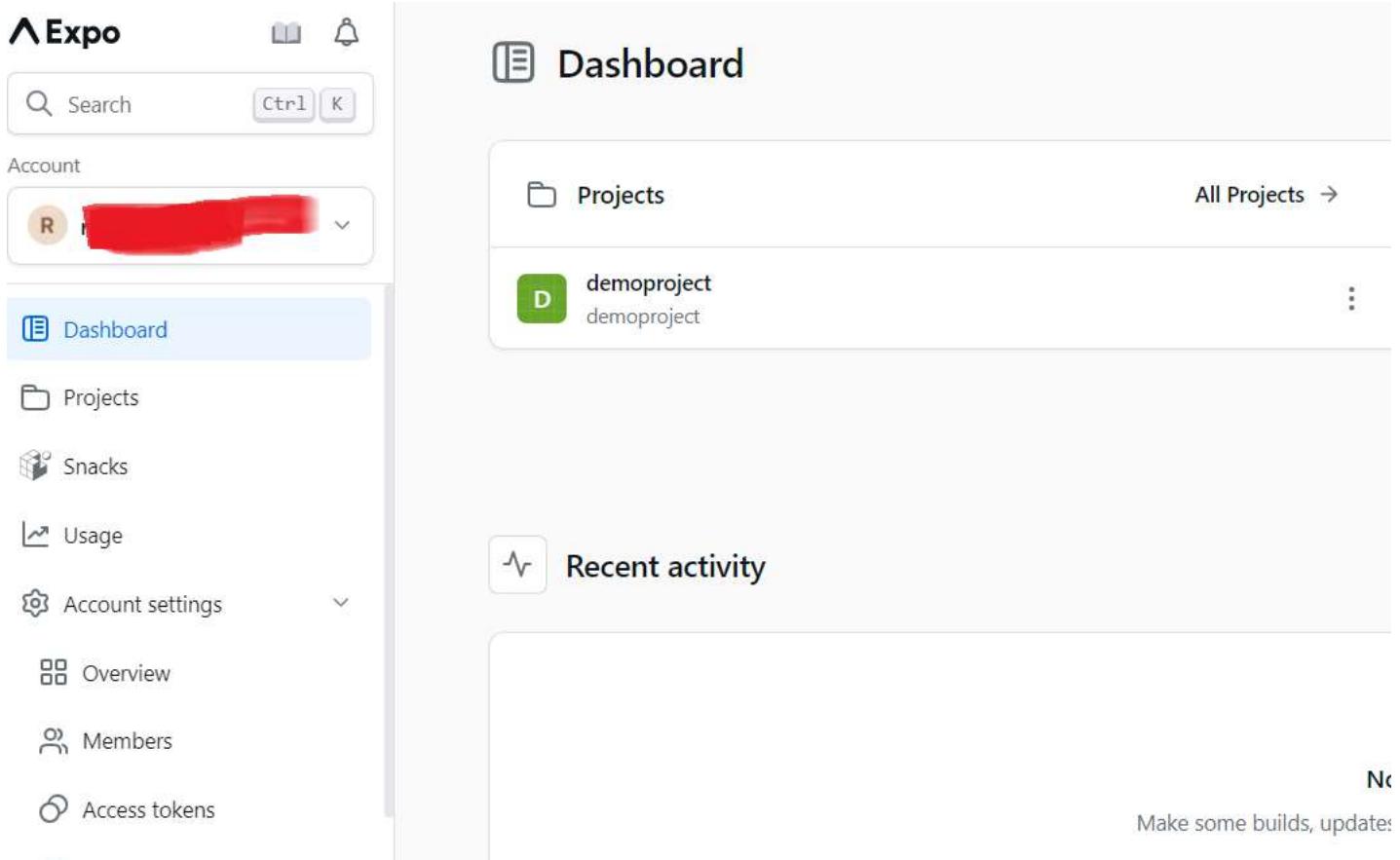
Run the following commands in your project root directory:



```
- npm install --global eas-cli
- eas init --id c5fa9ec3-f901-4add-b6b7-352ddbf3cc6f
```

[View project →](#)

8. Now, your Expo Dev dashboard will look like the screenshot below.



The screenshot shows the Expo Dev Dashboard interface. On the left, there is a sidebar with the following items:

- Search bar with 'Ctrl K' keyboard shortcut
- Account dropdown (redacted)
- Dashboard (selected)
- Projects
- Snacks
- Usage
- Account settings
- Overview
- Members
- Access tokens

The main dashboard area has the following sections:

- Dashboard** header
- Projects** section with a green folder icon and 'demoproject' listed. There is also a 'All Projects →' link and a three-dot menu icon.
- Recent activity** section with a note: 'No recent activity' and 'Make some builds, update:'.

9. Click the newly built project to see the details.



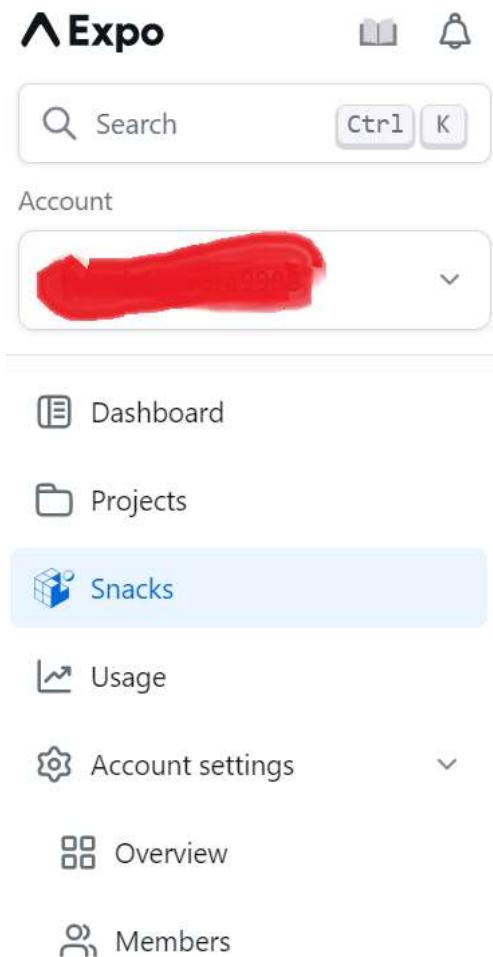
The screenshot shows the 'Overview' page for the 'demoproject' project. The page includes the following information:

- demoproject** (with a green 'D' icon)
- Owner (redacted)
- Slug: demoproject
- ID: c5fa9ec3-f901-4add-b6b7-352ddbf3cc6f

10. You will notice that the **ID** references shown in steps 5, 7, and 9 are the same.

Use Snack for a web-based development environment

1. You can also create React Native applications on Expo Dev's online platform, Expo Snack.
2. Expo Snack is a web-based development environment for Expo that lets you build and test React Native apps in your browser. You don't need to install any tools on your computer or phone to use Snack.
3. Now, in your left window pane, you will see a **Snacks** option. You need to click it.



4. After clicking **Snacks** your dashboard will look like the screenshot below. There, you need to click **New Snack**.



Create a Snack

Try out Expo in your browser and share your code with others using Snack

[Create a Snack ↗](#)

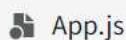
5. The dashboard for Snack will display as in the screenshot below. Here, you will see the default and basic layout of the code, which you will start to understand in upcoming hands-on labs.



smelly violet chip ⓘ

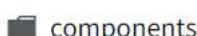
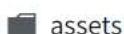
Not saved yet.

^ Open files



App.js

^ Project



```
1 import { Text, SafeAreaView, StyleSheet } from 'react-native'

2 // You can import supported modules from npm
3 import { Card } from 'react-native-paper';

4 // or any files within the Snack
5 import AssetExample from './components/AssetExample';

6
7 export default function App() {
8   return (
9     <SafeAreaView style={styles.container}>
10       <Text style={styles.paragraph}>
11         Change code in the editor and watch it change
12         get a shareable url.
13       </Text>
14       <Card>
15         <AssetExample />
16       </Card>
17     </SafeAreaView>
18   );
19 }
20

21 const styles = StyleSheet.create({
22   container: {
23     flex: 1,
24     justifyContent: 'center'.
25 }
```

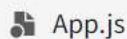
6. The basic file structure is on the left side of this window, and the different preview options are on the right side. These options allow you to easily view how your application will look on different devices.

7. To see the changes in the above screenshot, you can see the code for the **App.js** file. At line number 13, you need to replace this entirely with **hello world**, which will look like the screenshot below. Now, from this screenshot, you need to remove line numbers 15, 16, and 17.

**smelly violet chip** ⓘ

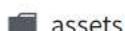
All changes saved less than 20 seconds ago. ✓

^ Open files



App.js

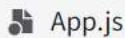
^ Project



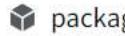
assets



components



App.js



package.json



README.md

```

1 import { Text, SafeAreaView, StyleSheet } from 'react-native'
2
3 // You can import supported modules from npm
4 import { Card } from 'react-native-paper';
5
6 // or any files within the Snack
7 import AssetExample from './components/AssetExample';
8
9 export default function App() {
10   return (
11     <SafeAreaView style={styles.container}>
12       <Text style={styles.paragraph}>
13         hello world
14       </Text>
15       <Card>
16         <AssetExample />
17       </Card>
18     </SafeAreaView>
19   );

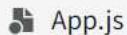
```

8. After removing line numbers 15, 16, and 17, it will look like the screenshot below.

**smelly violet chip** ⓘ

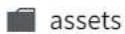
All changes saved less than a minute ago. ✓

^ Open files



App.js

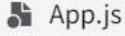
^ Project



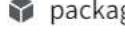
assets



components



App.js



package.json



README.md

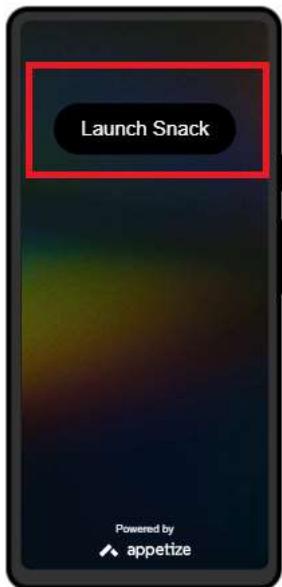
```

9 export default function App() {
10   return (
11     <SafeAreaView style={styles.container}>
12       <Text style={styles.paragraph}>
13         hello world
14       </Text>
15     </SafeAreaView>
16   );
17 }
18
19 const styles = StyleSheet.create({
20   container: {
21     flex: 1,
22     justifyContent: 'center',
23     backgroundColor: '#ecf0f1',
24     padding: 8,
25   },
26   paragraph: {
27     margin: 24,
28     fontSize: 18,
29     fontWeight: 'bold',
30     textAlign: 'center',
31   },
32 });
33

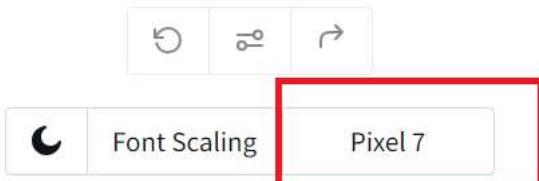
```

9. If you choose the **Android** tab, you will need to click **Launch Snack**, as shown in the screenshot below.

My Device **Android** iOS Web 



10. Select any device you want to see the output for. For example, click **Pixel 7**.



11. After clicking **Pixel 7**, you will see multiple device options, and you can select any of them.



12. After clicking **Launch Snack** you will see the output on the screen displaying **hello world**.

```
9  export default function App() {
10    return (
11      <SafeAreaView style={styles.container}>
12        <Text style={styles.paragraph}>
13          hello world
14        </Text>
15      </SafeAreaView>
16    );
17  }
18
19
20  const styles = StyleSheet.create({
21    container: {
22      flex: 1,
23      justifyContent: 'center',
24      backgroundColor: '#ecf0f1',
25      padding: 8,
26    },
27    paragraph: {
28      margin: 24,
29      fontSize: 18,
30      fontWeight: 'bold',
31      textAlign: 'center',

```

13. Similarly, you can choose **iOS** from the tabs and click **Launch Snack**. It will then ask you to open it in **Expo Go** by clicking **Open**.

My Device Android **iOS** Web



Font Scaling

iPhone 15 Pro

14. You will then see the **hello world** as the output. After clicking **iPhone 15 Pro**, you can also choose any other device.

My Device

Android

iOS

Web

[



Font Scaling

iPhone 15 Pro

15. Now select the **My Device** tab. It will display a scannable QR code, which you can scan with the Expo Go application installed on your mobile device.



My Device Android iOS Web

Download [Expo Go](#) and scan
the QR code to get started.



16. After scanning, the application will appear on your mobile device, allowing you to experience how it looks on mobile devices in real time.

Check Output From GitHub Repository

1. You can also check the output on **My Device**, **Android**, **iOS** and **Web** directly by importing code from the GitHub repository.
2. Now, import your GitHub repository by clicking the three dots and selecting **Import git repository**.

The screenshot shows the Snack web interface. On the left, there's a sidebar with 'Open files' (App.js) and 'Project' sections (assets, components, App.js, package.json, README.md). A red box highlights the three-dot menu icon in the Project section. A modal window titled 'Import files' is open, containing three options: 'Import git repository' (which is also highlighted with a red box), 'Import local file', and 'Export project'. The code editor on the right shows a snippet of App.js.

3. After clicking **Import git repository**, a pop-up will appear. Here, you need to paste your own GitHub repository link and then click on **Import repository**. This action will import the entire code as a Snack in this web development environment.

Import git repository

Import an Expo project from a public git repository. Note, files over 10MB are not imported.

Git URL

`https://github.com/ide/love-languages/tree/main/app`

[Show advanced options](#)

Import repository

4. After this, you can check the output by selecting either of **My Device**, **Android**, **iOS**, or **Web** options.

Author(s)

Richa Arora