Cheat Sheet: Expo Commands and Codes for Components and NavigationTop of Form

Download the **Cheat Sheet: Expo Commands and Codes for Components and NavigationTop of Form** document to access the Expo commands and codes, their descriptions, and some examples for the Expo commands/codes covered in this module.

Note: If the cheat sheet doesn't download, right-click and open the link in a new tab.

Expo Commands and Codes	Description	Command/Code Example

View Component	The View component is a core building block used to create a container for other components. It is analogous in web development and is primarily used for layout, styling, and grouping multiple elements like Text and Image.	
		```
		import React from 'react';
		import { View, Text, StyleSheet } from 'react-native';
		const MyComponent = () => {
		  return (
		    <View style={styles.container}>

		    </View>
		  );
		};
		```

Text Component	The Text component in React Native is used to display and style text content in an app.	
		```
		import React from 'react';
		import { Text, View } from 'react-native';
		const MyTextComponent = () => {
		  return (
		    <View>
		      <Text style={{ fontSize: 20 }}>Hello, React Native!</Text>
		      <Text style={{ color: 'blue' }}>This is a basic text component.</Text>
		    </View>
		  );
		};
		export default MyTextComponent
		```

Image Component	The Image component enables a user to display images in a particular application. It supports both local and remote image sources including custom styles.	
		```
		import React from 'react';
		import { Image, View, StyleSheet } from 'react-native';
		const MyImageComponent = () => {
		  return (
		    <View style={styles.container}>
		      <Image
		        source={{ uri: 'https://example.com/myimage.jpg' }}
		        style={styles.image}
		      />
		    </View>
		  );
		};
		export default MyImageComponent;
		```

Button Component	The Button component serves for rendering of a button which can be pressed using onPress prop.	
		```
		import React from 'react';
		import { Button, View } from 'react-native';
		const MyButtonComponent = () => {
		  return (
		    <View style={{ margin: 20 }}>
		      <Button
		        title="Press Me"
		        onPress={() => alert('Button Pressed!')}
		      />
		    </View>
		  );
		};
		export default MyButtonComponent;
		```

TextInput Component	The TextInput component in React Native is used to create an input field where users can enter text, commonly used for forms, search bars, or any kind of user input.	`import React, { useState } from 'react';`
		`import { TextInput, View, StyleSheet, Text } from 'react-native';`
		`const MyTextInputComponent = () => {`
		`  return (`
		`    <View style={styles.container}>`
		`      <TextInput`
		`        style={styles.input}`
		`        placeholder="Enter some text"`
		
		`      />`
		`      <Text>You typed: {text}</Text>`
		`    </View>`
		`  );`
		`};`
		`export default MyTextInputComponent;`

ScrollView Component	The ScrollView component in React Native is a container that allows users to scroll through a large set of content vertically or horizontally when it exceeds the screen size. It is typically used for layouts that need more flexibility with scrolling, like lists or long content.	`import { ScrollView, Text, StyleSheet } from 'react-native';`
		`const MyScrollViewComponent = () => {`
		`  return (`
		`    <ScrollView style={styles.scrollView}>`
		`      <Text style={styles.text}>Item 1</Text>`
		`      <Text style={styles.text}>Item 2</Text>`
		`      <Text style={styles.text}>Item 3</Text>`
		`      <Text style={styles.text}>Item 4</Text>`
		`      <Text style={styles.text}>Item 5</Text>`
		`      {/* More items can be added here */}`
		`    </ScrollView>`
		`  );`
		`};`
		`const styles = StyleSheet.create({`
		`  scrollView: {`
		`    marginHorizontal: 20,`
		`  },`
		`  text: {`
		`    fontSize: 20,`
		`    marginVertical: 10,`
		`  },`
		`});`
		`export default MyScrollViewComponent;`

FlatList	The FlatList component in React Native is an optimized component for rendering large lists of data. It efficiently renders only the visible items and dynamically loads more as the user scrolls, providing better performance for long lists compared to ScrollView.	`import { FlatList, Text, View, StyleSheet } from 'react-native';`
		`const MyFlatListComponent = () => {`
		`  const data = [`
		`    { key: 'Item 1' },`
		`    { key: 'Item 2' },`
		`    { key: 'Item 3' },`
		`    { key: 'Item 4' },`
		`    { key: 'Item 5' },`
		`  ];`
		`  return (`
		`    <FlatList`
		`      data={data}`
		`      renderItem={({ item }) => <Text style={styles.item}>{item.key}</Text>}`
		`    />`
		`  );`
		`};`
		`export default MyFlatListComponent;`

TouchableOpacity Component	This component in React Native is used to create a button that responds to user touch, providing a visual feedback effect by adjusting its opacity when pressed.	`import React from 'react';`
		`import { View, Text, TouchableOpacity, Alert } from 'react-native';`
		`const App = () => {`
		`  const handlePress = () => {`
		`    Alert.alert('Button Pressed!', 'You pressed the button.');`
		`  };`
		`  return (`
		`    <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>`
		`      <TouchableOpacity onPress={handlePress} style={{ padding: 10, backgroundColor: 'blue', borderRadius: 5`
		`        <Text style={{ color: 'white' }}>Press Me</Text>`
		`      </TouchableOpacity>`
		`    </View>`
		`  );`
		`};`
		`export default App;`

Install React Navigation Native Package	This command installs package which is essential for implementing navigation features in a React Native app using Expo.	`npx expo install @react-navigation/native`

Navigation Container	The NavigationContainer in React Native is like a container that holds all the screens/pages in your app and helps you move from one screen to another. It's needed to make navigation work properly.	`import { NavigationContainer } from '@react-navigation/native';`
		`export default function App() {`
		`  return (`
		`    <NavigationContainer>`
		`      {/* Other navigators or screens go here */}`
		`    </NavigationContainer>`
		`  );`
		`}`

Install React Native Screens and Safe Area Context.	It installs the react-native-screens and react-native-safe-area-context packages, which are used to optimize navigation performance and manage safe area insets in a React Native app.	`npx install react-native-screens react-native-safe-area-context`
Install React Native Gesture Handler	It installs the react-native-gesture-handler package, which provides enhanced gesture handling capabilities for building touch-enabled interfaces in React Native apps.	`npx expo install react-native-gesture-handler`
Stack Navigator Package for Expo CLI	The command installs the @react-navigation/native-stack package, which provides a stack navigator for managing screen transitions in a React Native app using Expo.	`npx expo install @react-navigation/native-stack`

Stack Navigator	The Stack Navigator in React Navigation is a navigation component that allows users to navigate between screens in a stack-like manner, where each new screen is pushed on top of the previous one, enabling a back navigation feature.	`import React from 'react';`
		`import { NavigationContainer } from '@react-navigation/native';`
		`import { createNativeStackNavigator } from '@react-navigation/native-stack';`
		`import { Button, Text, View } from 'react-native';`
		`const Stack = createNativeStackNavigator();`
		`function HomeScreen({ navigation }) {`
		`  return (`
		`    <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>`
		`      <Text>Home Screen Using Stack Navigator</Text>`
		`      <Button`
		`        title="Go to Details"`
		`        onPress={() => navigation.navigate('Details')}`
		`      />`
		`    </View>`
		`  );}`
		`function DetailsScreen() {`
		`  return (`
		`    <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>`
		`      <Text>Details Screen Using Stack Navigators</Text>`
		`    </View>`
		`  );}`
		`export default function App() {`
		`  return (`
		`    <NavigationContainer>`
		`      <Stack.Navigator initialRouteName="Home">`
		`        <Stack.Screen name="Home" component={HomeScreen} />`
		`        <Stack.Screen name="Details" component={DetailsScreen} />`
		`      </Stack.Navigator>`
		`    </NavigationContainer>`
		`  );}`

Drawer Navigator package for Expo CLI	The command installs the @react-navigation/drawer package, which enables the use of a drawer navigation interface in a React	`npx expo install @react-navigation/drawer`

	Native app using Expo.	

Drawer Navigator	The Drawer Navigator in React Navigation is a navigation component that provides a side menu drawer for navigating between different screens in a React Native app, typically accessed by swiping or tapping a menu button.	`import React from 'react';`
		`import { Button, Text, View } from 'react-native';`
		`import { NavigationContainer } from '@react-navigation/native';`
		`import { createDrawerNavigator } from '@react-navigation/drawer';`
		`const HomeScreen = ({ navigation }) => (`
		`  <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>`
		`    <Text>Home Screen Using Drawer Navigator</Text>`
		`    <Button`
		`      title="Open Drawer"`
		`      onPress={() => navigation.openDrawer()}`
		`    />`
		`  </View>`
		`);`
		`const SettingsScreen = () => (`
		`  <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>`
		`    <Text>Settings Screen Using Drawer Navigator</Text>`
		`  </View>`
		`);`
		`const Drawer = createDrawerNavigator();`
		`function App() {`
		`  return (`
		`    <NavigationContainer>`
		`      <Drawer.Navigator initialRouteName="Home">`
		`        <Drawer.Screen name="Home" component={HomeScreen} />`
		`        <Drawer.Screen name="Settings" component={SettingsScreen} />`
		`      </Drawer.Navigator>`
		`    </NavigationContainer>`
		`  );`
		`}`
		`export default App;`

Tab Navigator Package for Expo CLI	This command installs the package, which allows the implementation of a bottom tab navigation interface in a React Native app using Expo.	`npx expo install @react-navigation/bottom-tabs`

Tab Navigator	A Tab Navigator is a component in React Native that allows users to switch between different screens or sections of an app using tabs, typically displayed at the bottom or top of the screen.	`import React from 'react';`
		`import { Text, View } from 'react-native';`
		`import { NavigationContainer } from '@react-navigation/native';`
		`import { createBottomTabNavigator } from '@react-navigation/bottom-tabs';`
		`const HomeScreen = () => (`
		`  <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>`
		`    <Text>Welcome to Home Screen</Text>`
		`  </View>`
		`);`
		`const SettingsScreen = () => (`
		`  <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>`
		`    <Text>Settings Screen1</Text>`
		`  </View>`
		`);`
		`const Tab = createBottomTabNavigator();`
		`function App() {`
		`  return (`
		`    <NavigationContainer>`
		`      <Tab.Navigator>`
		`        <Tab.Screen name="Home" component={HomeScreen} />`
		`        <Tab.Screen name="Settings" component={SettingsScreen} />`
		`      </Tab.Navigator>`
		`    </NavigationContainer>`
		`  );`
		`}`
		`export default App;`